NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

SHIP OUTLINE FEATURE SELECTION
USING B-SPLINE FUNCTION

by

Werawong Thavamongkon

December 1984

Thesis Advisor: Chin-Hwa Lee

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Ship Outline Feature Selection Using B-spline Function		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; December 1984
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Werawong Thavamongkon		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		12. REPORT DATE December 1984
		13. NUMBER OF PAGES 141
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

B-spline function, ship outline feature, feaure extraction, Fourier transformcoefficients

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis presents two methods of ship classification with IR images. The first method is the Fourier Coefficient method which transform the sample points of a superstructure profile of a ship to the spatial frequency components. The shape of the coefficient curve can be used to classify the type of a ship from 8 different categories. But, the differences is so minor that it is difficult to implement a

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

computer program to recognize it. The second method is a B-spline Coefficient method which uses the uneven spaced spline coefficients to find the beginning, the peak, and the area of the lumps of a ship for classification. This method is better than the Fourier Coefficient method. The study of These methods is presented here.

Ship Outline Feature Selection Using B-spline Function

by

Werawong Thavamongkon
Lieutenant, Royal Thai Navy
B.S., Royal Thai Naval Academy, 1975

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1984

ABSTRACT

This thesis presents two methods of ship classification with IR images. The first method is the Fourier Coefficient method which transform the sample points of a superstructure profile of a ship to the spatial frequency components. The shape of the coefficient curve can be used to classify the type of a ship from 8 different categories. But, the differences is so minor that it is difficult to implement a computer program to recognize it. The second method is a B-spline Coefficient method which uses the uneven spaced spline coefficients to find the beginning, the peak, and the area of the lumps of a ship for classification. This method is better than the Fourier Coefficient method. The study of These methods is presented here.

TABLE OF CONTENTS

I. INTRODUCTION 12

II. PREPROCESSING 13
A. DATA COLLECTION 13
B. SOBEL OPERATOR 14
C. THE USE OF A SOBEL OPERATOR 16
D. EDGE THRESHOLD STRATEGIES 18
E. HOW TO EXTRACT THE PROFILE 21
F. HOW TO OBTAIN THE SUPERSTRUCTURE 21
G. CONTOUR FOLLOWING 22
H. CLOSING OPERATION 27
I. PROFILE ROTATION 29

III. FOURIER COEFFICIENT METHOD 31
A. VARIATION OF SHIP SUPERSTRUCTURE WITH RANGE 43
1. Background 43
2. Remark 46

IV. B-SPLINE COEFFICIENT METHOD 48
A. BACKGROUND 48
B. B-SPLINE APPROXIMATION WITH FREE KNOT 49
1. Interpolation and Approximation Using B-spline Function 51
C. TO DETERMINE THE KNOT POSITION AND THE B-SPLINE COEFFICIENTS 52
1. Limitation on B-spline Coefficient Determination 53
2. Selection of Smoothing Factor Value(S) . . 58
3. The Output Cx,Cy and Knots Profiles . . . 67

D. SHIP DESCRIPTION 76
E. SHIP CLASSIFICATION 79
1. Lump Detection 80
2. To Determine the Area Under a Lump 87
F. CONSTRUCTION OF THE DECISION TREE 88
G. SUMMARY 100

V. CONCLUSION 103

APPENDIX A: THE PROGRAM TO OBTAIN THE SUPERSTRUCTURE . 108

APPENDIX B: THE PROGRAM OF CONTOUR FOLLOWING 113

APPENDIX C: THE SUBROUTINE "PARAM" 117

APPENDIX D: THE PROGRAM TO FIND B-SPLINE COEFFICIENT . 130

LIST OF REFERENCES 140

INITIAL DISTRIBUTION LIST 141

LIST OF TABLES

I. Logarithmic Magnitude of Ships 42
II. Range Estimation 46
III. Comparison of Different Types of Ships 98

LIST OF FIGURES

2.1 Sobel Operator 16
2.2 A CGN at a Range of 45000 feet 16
2.3 Image from a Sobel Operator 17
2.4 Blured Image of a LST at a Range 67000 feet 17
2.5 Noisy Image from a Sobel Operator 17
2.6 Histogram of Figure 2.3 19
2.7 Cumulative Distribution of the Histogram in Figure 2.6 20
2.8 Silhouette of a CGN in Figure 2.3 after Threshold 21
2.9 Superstructure Profile of Figure 2.8 22
2.10 The 3 by 3 Kernel 24
2.11 The Step Checks in the North Direction 24
2.12 The Step Checks in the East Direction 25
2.13 The Step Checks in the South Direction 25
2.14 The Step Checks in the West Direction 25
2.15 Flow Chart of Contour Following 26
2.16 Contour Image of a CGN at a Range of 45000 feet . . 27
2.17 The Process of Dilation 28
2.18 The Profile after Dilation and Erosion (Closing Process) 29
2.19 Rotation Process 30
2.20 Rotated Profile of a CGN at a Range of 45000 feet 30
3.1 Logarithmic Magnitude of the Fourier Transform of a DD 34
3.2 Logarithmic Magnitude of the Fourier Transform of a Container 35

3.3 Logarithmic Magnitude of the Fourier Transform of a Freighter 36
3.4 Logarithmic Magnitude of the Fourier Transform of a AOR 37
3.5 Logarithmic Magnitude of the Fourier Transform of a LST 38
3.6 Logarithmic Magnitude of the Fourier Transform of a FF 39
3.7 Logarithmic Magnitude of the Fourier Transform of a CGN 40
3.8 Logarithmic Magnitude of the Fourier Transform of a DDG 41
3.9 The System of the Camera 43
3.10 Range Calculation 44
4.1 Plot of the Original Data and the Approximate with Free Knot 52
4.2 Knot Selection if the NEST Parameter is too Small 53
4.3 A CGN at Range of 45000 feet 55
4.4 A CGN at Range of 55000 feet 56
4.5 A CGN at Range of 64000 feet 57
4.6 Plot N vs S for a CGN 59
4.7 Plot N vs S for a DD 60
4.8 Plot N vs S for a Container 61
4.9 Plot N vs S for a Freighter 62
4.10 Plot N vs S for a AOR 63
4.11 Plot N vs S for a LST 64
4.12 Plot N vs S for a FF 65
4.13 Plot N vs S for a DDG 66
4.14 Plot X,Y,Cx, and Cy vs Z for a FF at a Range of 43 K-ft 68
4.15 Plot X,Y,Cx, and Cy vs Z for a LST at a Range of 51 K-ft 69

4.16 Plot X,Y,Cx, and Cy vs Z for a CGN at a Range of 45 K-ft 70
4.17 Plot X,Y,Cx, and Cy vs Z for a DDG at a Range of 41 K-ft 71
4.18 Plot X,Y,Cx, and Cy vs Z for a DD at a Range of 77 K-ft 72
4.19 Plot X,Y,Cx, and Cy vs Z for a AOR at a Range of 78 K-ft 73
4.20 Plot X,Y,Cx, and Cy vs Z for a Freighter at a Range of 40 K-ft 74
4.21 Plot X,Y,Cx, and CY vs Z for a Container at a Range of 28 K-ft 75
4.22 Frigate 76
4.23 Tank Landing Ship(LST) 77
4.24 Guided Missile Destroyer(DDG) 78
4.25 Destroyer 78
4.26 Guided Missile Cruiser(CGN) 78
4.27 Plot X,Y vs Z for a CGN at 45kft 82
4.28 Plot Cx,Cy vs Z for a CGN at 45kft 83
4.29 Flow Chart for Detecting Lumps 86
4.30 First Procedure to Determine the Area 87
4.31 Step by Step Procedure to Determine the Area . . . 88
4.32 Plot the Beginning, Area, and Peak of a FF 90
4.33 Plot the Beginning, Area, and Peak of a LST 91
4.34 Plot the Beginning, Area, and Peak of a DD 92
4.35 Plot the Beginning, Area, and Peak of a DDG 93
4.36 Plot the Beginning, Area, and Peak of a CGN 94
4.37 Plot the Beginning, Area, and Peak of a AOR 95
4.38 Plot the Beginning, Area, and Peak of a Freighter 96
4.39 Plot the Beginning, Area, and Peak of a Container 97
4.40 The Decision Tree 99

4.41 Noisy Image 100
4.42 High Threshold 101
4.43 Low Threshold 101
4.44 Before Closing 102
4.45 After Closing 102
5.1 Edge Image of a DD at a Range of 79000 feet . . . 104
5.2 Before Closing Process 104
5.3 After Closing Process 104
5.4 Logarithmic Magnitude of a CGN at a Range of 45000 feet 105
5.5 Logarithmic Magnitude of a CGN at a Range of 55000 feet 106

I. INTRODUCTION

It is quite important to classify the ships according to their types. Classification can be done in a number of different ways. The simplest is the visual method that is prone to error. Other methods for classification are the Fourier Coefficient method and the B-spline Coefficient method. In these methods the necessary information for classification can be obtained from the superstructure profile.

The Fourier Coefficient method samples the superstructure profile at every chosen points. The function values at the sampling points are transformed into the spatial components. The logarithm of the magnitude of these components is plotted and compared with the standard plot to recognize the type of the ship.

In the B-spline Coefficient method, the spline coefficients along the X axis and the Y axis are used to reconstruct the superstructure profile. The shape of the curve of the spline coefficients is in some way similar to the shape (the position of the lumps) of the superstructure profile. The ship classification may be achieved by recognizing the beginning, the peak, and the area of the lumps.

II. PREPROCESSING

Preprocessing is the procedure to obtain the superstructure profile of a ship from the IR image. Then, Fourier coefficient or spline coefficient methods can be used. The details of the preprocessing procedures are a follows.

A. DATA COLLECTION

The data consists of the IR image of eight different types of ships.

1. DD - Destroyer; "HALL" class.
2. Container; The class is unidentified.
3. Freighter; The class is unidentified.
4. AOR - Replenishment oiler; "WICHITA" class.
5. LST - Tank landing ships; "NEWPORT" class.
6. FF - Frigate; "GARCIA" class.
7. CGN - Guided missile cruiser (Nuclear propulsion); "BAINBRIDGE" class.
8. DDG - Guided missile destroyer; "CHARLES F. ADAMS" class.

These images are taken from an aircraft which is flown at a height of 500 feet with a speed of 400 knots toward the side of the ship. The aspect angles for these images are 90 degrees which may be slightly off in some images. The inaccuracy of the aspect angle arises from the fact that the photos are taken while the aeroplane keeps on moving. All data of the images are stored in a digital magnetic tape with 256 by 64 bytes per image. Thus, the number of bytes required for each image is 16384 and each byte represents the intensity of a pixel. For each record of the image, a label is coded in the last 8 bytes as follows:

1. Byte (16377) = Run number in each flight which passes

the ship.

2. Byte (16378) = Video tape time code when the data is taken; in minutes.
3. Byte (16381) = Video tape time code; in seconds.
4. Byte (16382) = Video tape time code; in thirtieth of a second.
5. Byte (16379) = Range in kilo-feet which is the distance measured from the radar it may have an error 1 to 2 kilo-feet.
6. Byte (16380) = Aspect angle; degrees from the bow of the ship.
7. Byte (16383) = Ship class.
8. Byte (16384) = ID, 1 = for training, 2, 3, 4, 5 = for testing.

The run number and the time code together uniquely define a specific image that represents a single TV frame with no averaging. In addition, the time interval between the end of one image to the beginning of the other is approximately 1.5 seconds. Also, there are inherent random noise in the record which arises from the photo instrument and the process of storing them on to the digital magnetic tape.

B. SOBEL OPERATOR

The Sobel Operator technique is used to find the edge. To determine the edge, the Sobel Operator uses the difference of gray levels of the pixels in a 3 by 3 matrix as shown in Figure 2.1.

a,b,c,d,e,f,g,h, and i are the values of the gray levels at the position of (x-1,y), (x,y-1), (x-1,y), (x,y), (x+1,y), (x-1,y+1), (x,y+1), and (x+1,y+1) respectively. The Laplacian estimate is defined as

$$\frac{\partial^2 f}{\partial x^2} \simeq f(x,y) - f(x+1,y) - \left[f(x+1,y) - f(x+2,y)\right] \qquad (2.1)$$

The basis vector for all directions are (a-2b+c), (g-2h+i), (a-2d+g), and (c-2f+i). Furthermore, each of the basis vector is convolved with the image as follows:

along the x axis

$$d_x = \left[f(x-1,y-1) + 2f(x,y-1) + f(x+1,y-1)\right] - \left[f(x-1,y+1) + 2f(x,y+1) + f(x+1,y+1)\right] \tag{2.2}$$

along the y axis

$$d_y = \left[f(x+1,y-1) + 2f(x+1,y) + f(x+1,y+1)\right] - \left[f(x-1,y-1) + 2f(x-1,y) + f(x-1,y+1)\right] \tag{2.3}$$

Since the magnitude of the resulting vector is the absolute value of the convolved results, the edge magnitude S(x,y) [Ref. 1],

$$S(x,y) = (d_x^2 + d_y^2)^{1/2} \tag{2.4}$$

Note that the Sobel operator does not use the gray level at the position (x,y). The advantage of using a Sobel operator over others is that the resulting edge is smoother due to a 3 by 3 matrix approach. If we compare the Sobel operator with the Laplacian operator, it is seen that the Sobel operator using the four basis vector as shown above will provide more accurate reading because of noise reduction in the original image. Hence, The Sobel operator is often used in the preprocessing operation.

a	b	c
d	e	f
g	h	i

Figure 2.1 Sobel Operator.

C. THE USE OF A SOBEL OPERATOR

When a Sobel operator is used at the edge of the image frame, the pixel level which lies out of the frame will be set equal to that of the adjacent pixel within the frame. The original image of a guided missile cruiser is shown in Figure 2.2. Since the result of the Sobel operator is numerically greater than 8 bits range, we have to rescale the result back to 8 bits range. This is achieved by determining the maximum and the minimum of the gray level. They are then used to rescale the gray level in the results of the Sobel operator as shown in Figure 2.3. In some instances, the original image is very poor as shown in Figure 2.4. Attempts to determine the edge of this image failed as shown in Figure 2.5.

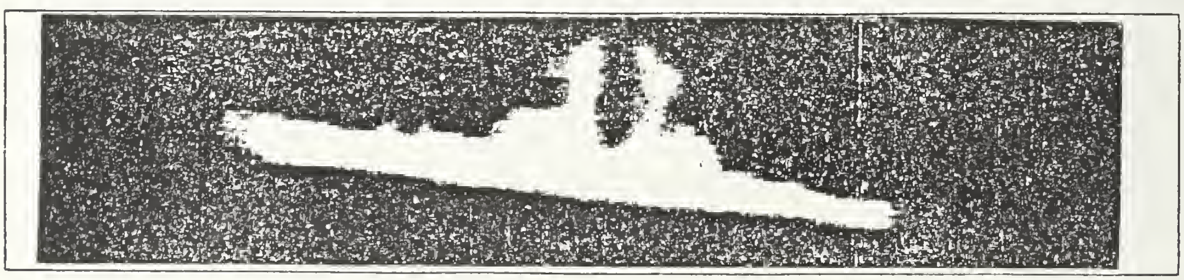


Figure 2.2 A CGN at a Range of 45000 feet.

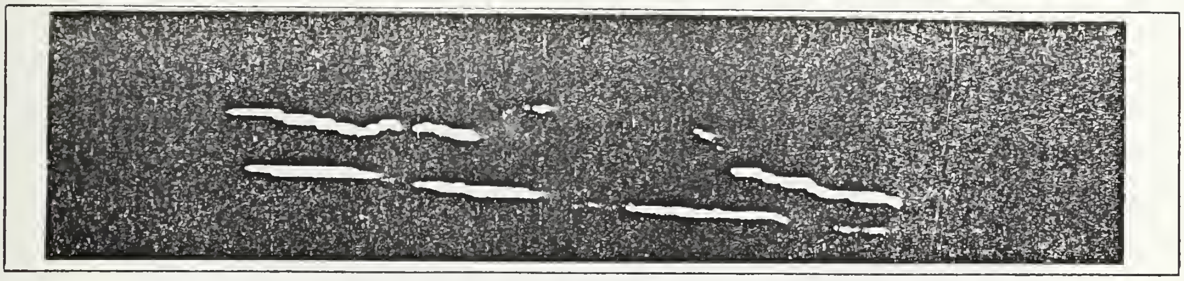


Figure 2.3   Image from a Sobel Operator.

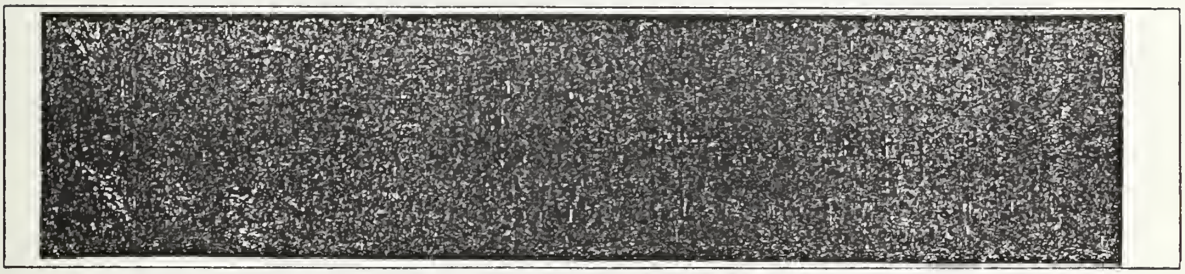


Figure 2.4   Blured Image of a LST at a Range 67000 feet.

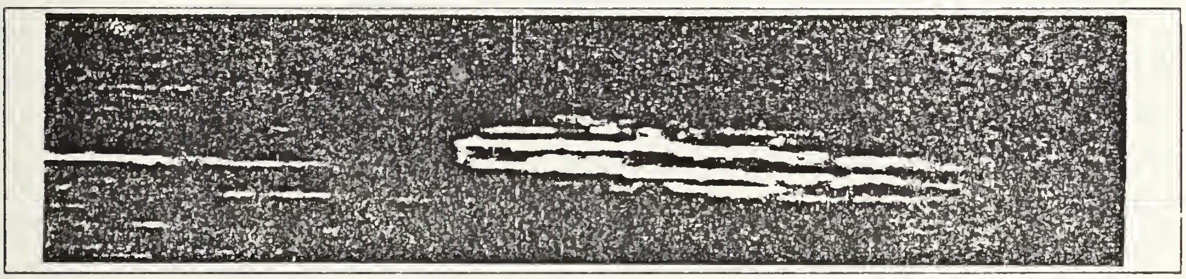


Figure 2.5   Noisy Image from a Sobel Operator.

D. EDGE THRESHOLD STRATEGIES

Edge threshold strategies are used to extract the edge profiles from the Sobel results. In this case, we use only the pronounced value of an edge element at x if g(x) is greater than certain threshold value [Ref. 2].

$$G(x,y) \begin{cases} = g(x,y) & \text{if } g(x\ y) \geq \text{threshold} \\ = 0 & \text{otherwise} \end{cases} \tag{2.5}$$

To increase the contrast of the image to a silhouette form, G(x,y) is defined as

$$G(x,y) \begin{cases} = 255 & \text{if } g(x,y) \geq \text{threshold} \\ = 0 & \text{otherwise} \end{cases} \tag{2.6}$$

The choice of the threshold value is based upon the histogram of the edge image as shown in Figure 2.6. The estimated critical gray level is chosen so that a majority number of the pixels with value between 0 to 255 will fall below the critical value. Alternatively, histogram equalization may be used to determine the desired threshold level as shown in Figure 2.7. In this case, trial and error method was used in conjunction with the above method to obtain the threshold so that the correct profile is ascertained.

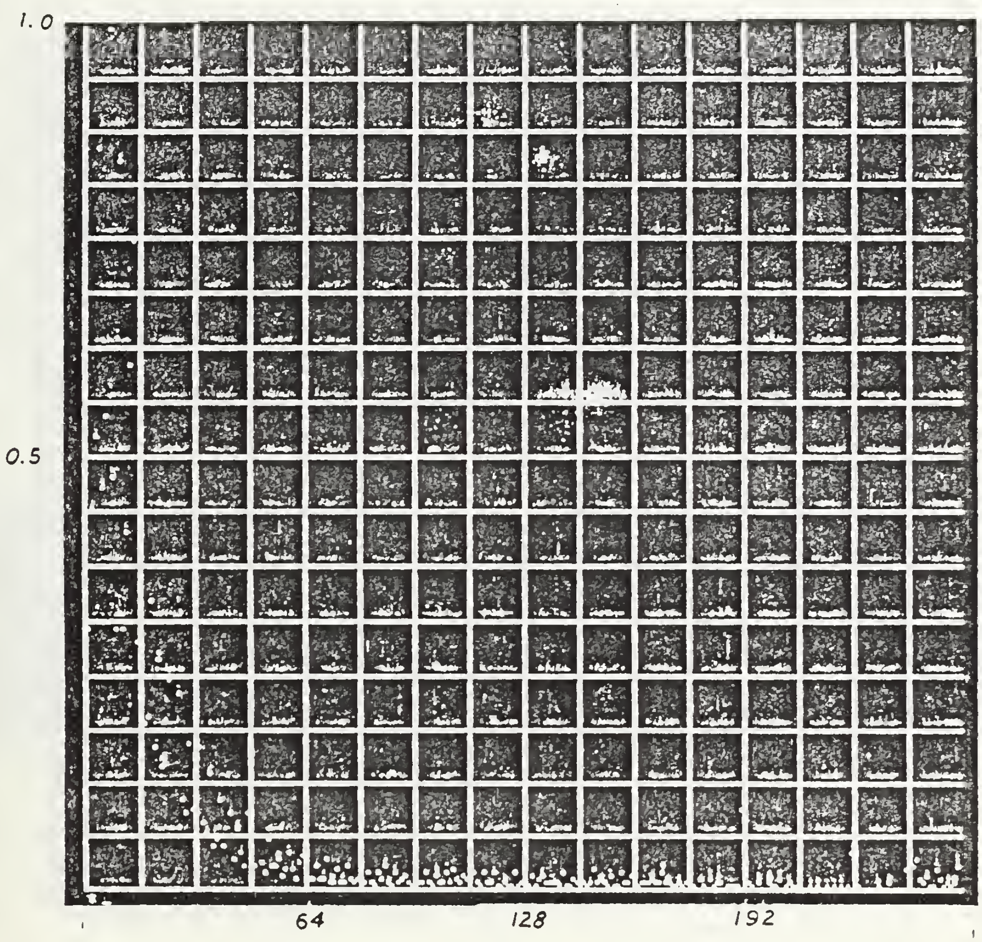


Figure 2.6 Histogram of Figure 2.3.

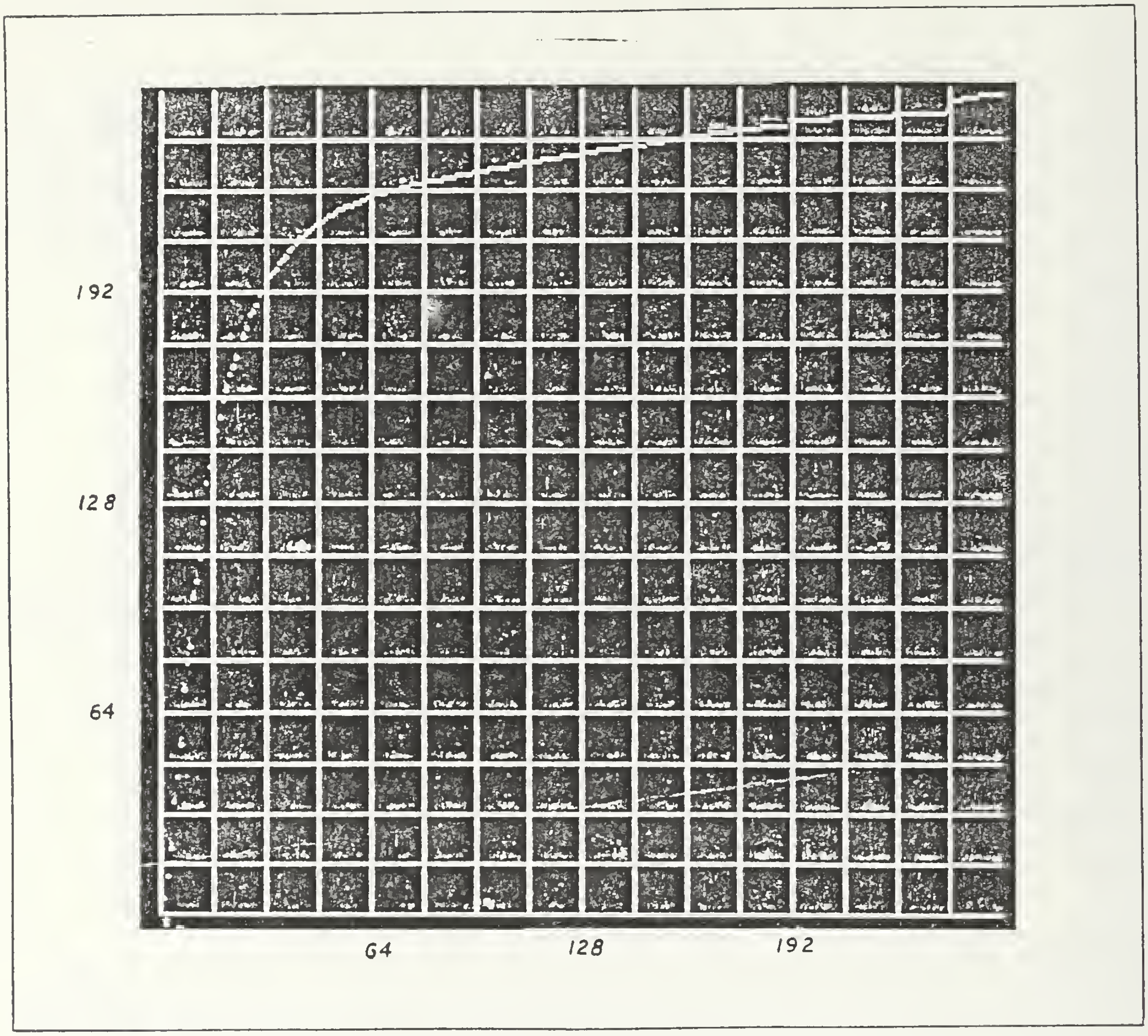


Figure 2.7 Cumulative Distribution of the Histogram in Figure 2.6.

E. HOW TO EXTRACT THE PROFILE

The edge image of the guided missile cruiser shows little variations in the gray level as shown in Figure 2.3. These variations are caused by the noise in the original image. In this case, the choice of the threshold value is based upon both the histogram and the cumulative distribution of the edge image so that it contains 90 % of the pixels. Therefor, the chosen gray level is 110 and the result is shown in Figure 2.8.

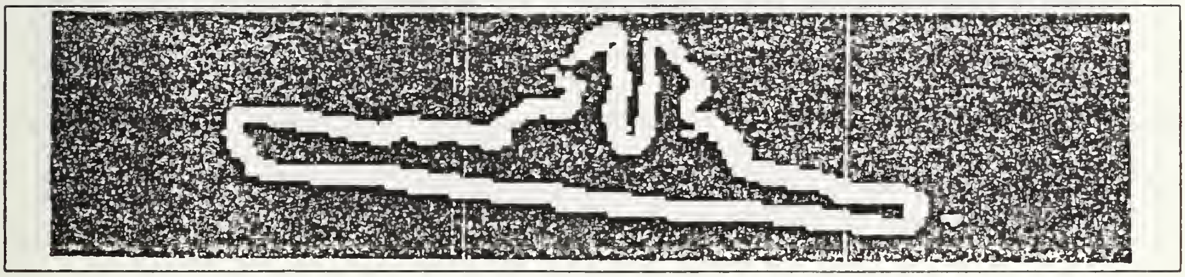


Figure 2.8 Silhouette of a CGN in Figure 2.3.

F. HOW TO OBTAIN THE SUPERSTRUCTURE

The original image of the ship is taken from the aeroplane with different displacement from waterline to the superstructure in a rough sea, so that the profile of the ship with respect to the sea surface varies. Thus, we have to eliminate some information in the image of the ship by considering the superstructure only. In considering the overall ship structure, it is obvious that the largest distance is between the bow and stern span. Therefore, the bow and stern points are located first in the program as shown in Appendix A. Then we consider the slope of the bow and stern of the ship and set all the gray values below the

slope line to zero. Hence, it results in a superstructure of the ship as shown in Figure 2.9. However, in some images, there is a lot of noise in the background which cause difficulty in locating the bow and stern. Under these circumstances, the dimensions of a ship are estimated by trial and error method. Then the gray level which lies outside is set to zero and an estimate of the bow and stern slope can be made.

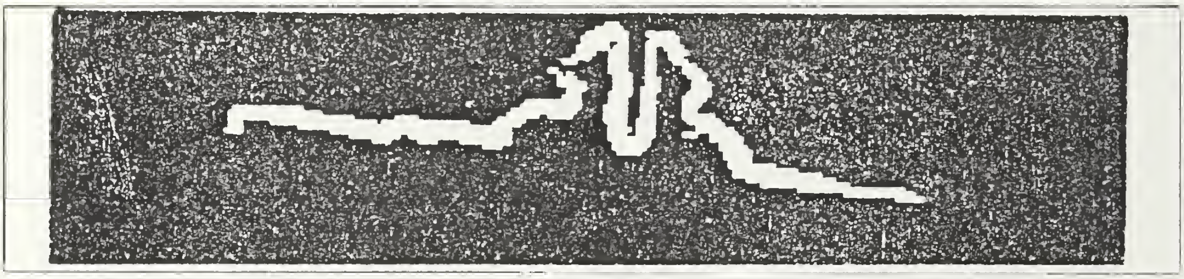


Figure 2.9 Superstructure Profile of Figure 2.8.

G. CONTOUR FOLLOWING

The noise in the background of the original image, yields wide edge structure. In considering this factor, the contour following procedure tracking the inner part of the image in Figure 2.9 gives the superstructure profile. The objective of this contour following is to describe the bow and stern points of the ship, the direction, and the position of the edge of the superstructure. The contour tracing is done in the counter-clockwise direction which compares pixel value of 0 or 255 in a 3 by 3 matrix in the following manner.

Starting from the leftmost point in the superstructure image in Figure 2.9 the contour profile tracing is accomplished by examining the neighbors of a 3 by 3 kernel located at the curser position as shown in Figure 2.10. The curser is moved along the profile. All successive positions of the curser constitute the contour profile of the ship. The testing procedure is explained below.

1. Initialize the curser position to the beginning of the thresholded image with the gray level of 255 and the estimate direction of the next position.
2. Check for reaching the end of the profile, if it is at the ending of the profile then stop, if not go to 3.
3. Check for the estimate to see whether it is in the direction of, North, East, South, or West. If the direction it is North then go to 4, if it is East then go to 5, if it is South then go to 6, and if it is West then go to 7.
4. Determine the next position with the gray value of 255 in the counter-clockwise direction as shown in Figure 2.11. Store the position found and move the curser to that position. Update the estimate direction to the one last found; then go to 2.
5. The procedure is the same as that in step 4 except search pattern is shown in Figure 2.12.
6. The procedure is the same as that in step 4 except search pattern is shown in Figure 2.13.
7. The procedure is the same as that in step 4 except search pattern is shown in Figure 2.14.

The flow chart of the procedure is shown in Figure 2.15, and the detail of each procedure are included in Appendix B. The testing procedure have to be performed in such a maner that the resulting contour is a good representation of the superstructure line. The result of the contour

image is shown in Figure 2.16. If the superstructure in Figure 2.9 has wide edge, we can not find the contour image. We have to use the additional step which is called the "Closing" operation.

Figure 2.10 The 3 by 3 Kernel.

4	3	2
5		1
6	7	

Figure 2.11 The Step Checks in the North Direction.

6	5	4
7		3
	1	2

Figure 2.12   The Step Checks in the East Direction.

	7	6
1		5
2	3	4

Figure 2.13   The Step Checks in the South Direction.

2	1	
3		7
4	5	6

Figure 2.14   The Step Checks in the West Direction.

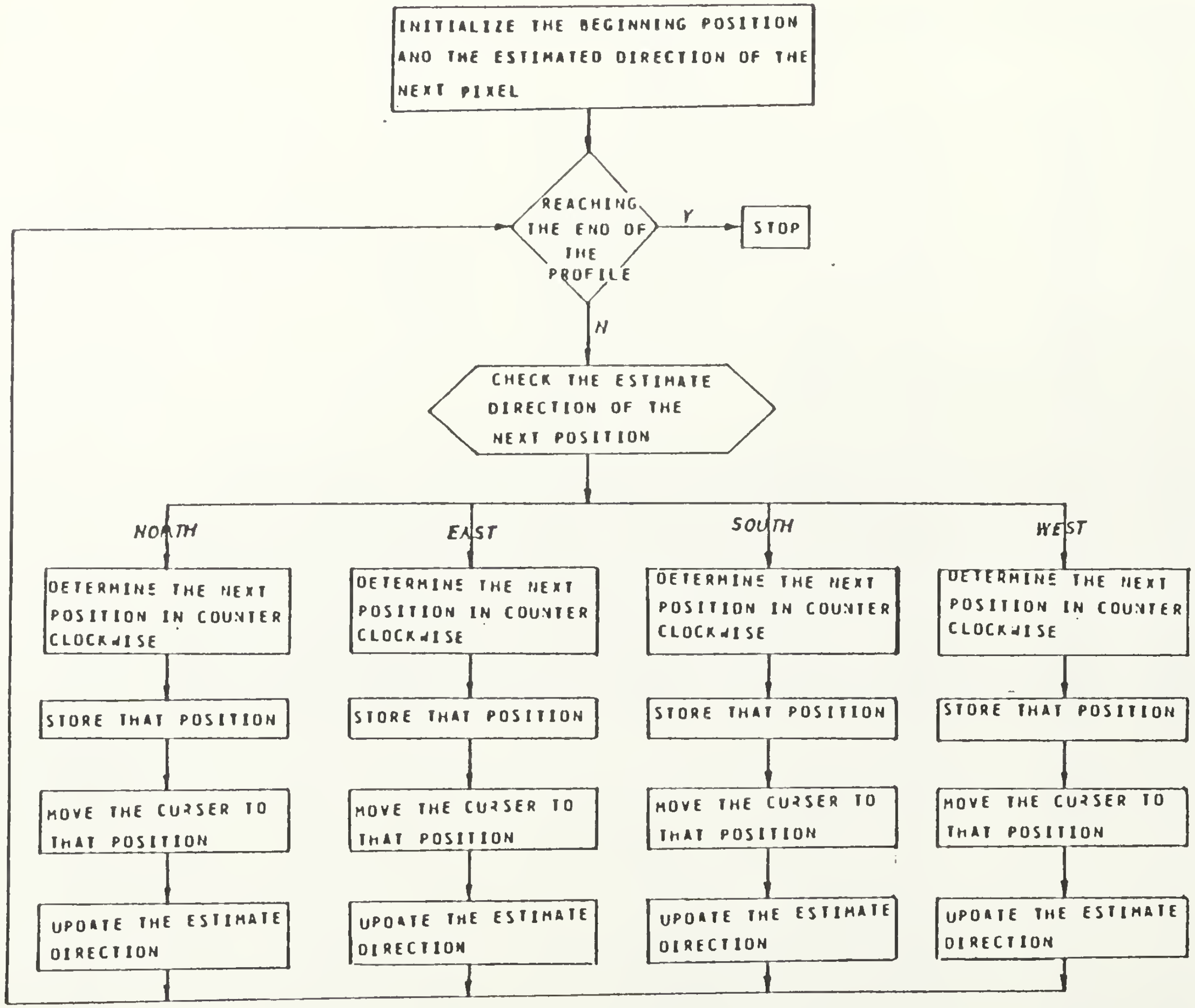


Figure 2.15 Flow Chart of Contour Following.

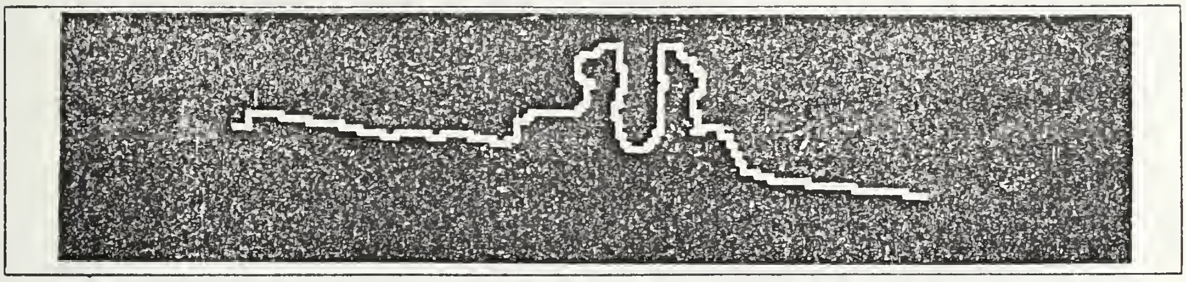


Figure 2.16 Contour Image of a CGN at a Range of 45000 feet.

H. CLOSING OPERATION

Some results from the superstructure extraction process are discontinuous because the gray level of those areas of the structure is less than the threshold value. If we decrease the threshold value, the details of the superstructure are effected. It is necessary to use the "Closing" operation which consists of the "dilation" process to smooth the superstructure profile used the "Erosion" process. All direction are dilated similarly which causes an smoothing effect on the edges, The superstructure increases in total area. Then, use the "Erosion" process to shrink (subtract) the dilated part in all directions, thus obtain the smoothed superstructure with the appropriate size. The Closing process employs the dilation process which yields an output image from an input image. The Dilation results is shown in Figure 2.17.

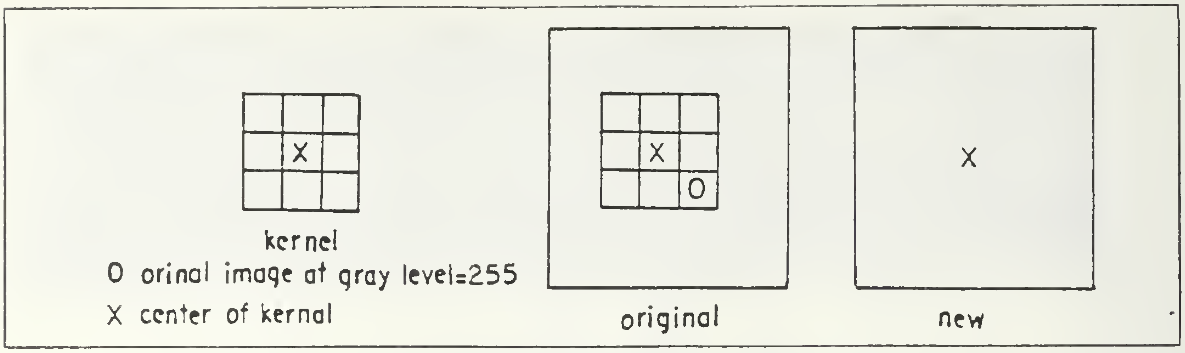


Figure 2.17 The Process of Dilation.

We examine the gray value of each pixel in the original image. Begining from the pixel at the first row and the first column. The procedures are the following.

1. Considering one pixel in the original image with the kernel (B) centered there. If at least one pixel in the kernel has a value of 255, we let the gray level of the output image at the center of the kernel be 255.
2. We shift the center of the kernel 1 column to the right. Then following the same procedure as in step 1 until the last column is reached.
3. We shift the position of the kernel to the next row and starting from the first column. Then, following the same procedure as in step 1 and step 2 until the last row and the last column is reached.

The result obtained from the dilation process is an image with enlarged structure. The second procedure in the Closing operation is the Erosion process. The Erosion process perform the same procedures as the dilation process except for step 1. If every pixel in the kernel are 255, we let the gray level in the new image at the center of the kernel be 255. Otherwise, it will be 0. The output image obtained will have smooth edge with minor change occurring

in the edge detail as shown in Figure 2.18. In this case, we use an kernel of size 3 by 3. If we increase the size of the kernel, the details of the image are decreased.

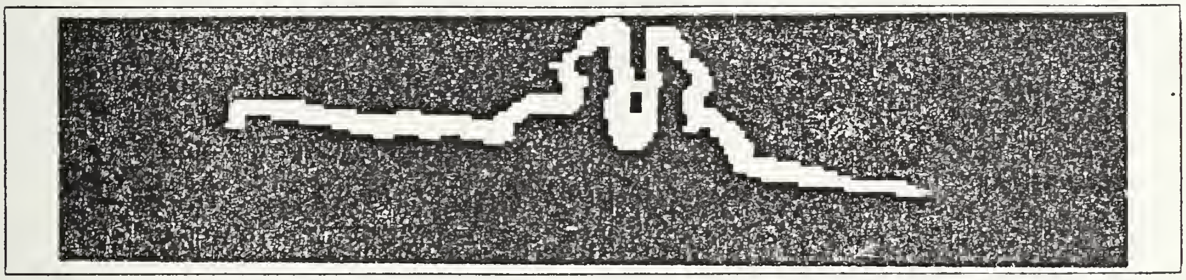


Figure 2.18 The Profile after Dilation and Erosion (Closing Process).

I. PROFILE ROTATION

Often in the contour image, the first and the last point of the superstructure are not at the same horizontal level. We have to rotate the contour image by setting the two points to the same horizontal level. How to rotate it from the Y, X axis to the Y', X' axis is shown in Figure 2.19.

If the angle value θ is positive, it will be

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix}$$

If the angle value θ is negative, it will be

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix}$$

For some of the contour profile, part of the profile after rotation will be out of the image frame. Then we have to shift this contour down by 20 pixels position. The rotated profile is shown in Figure 2.20.

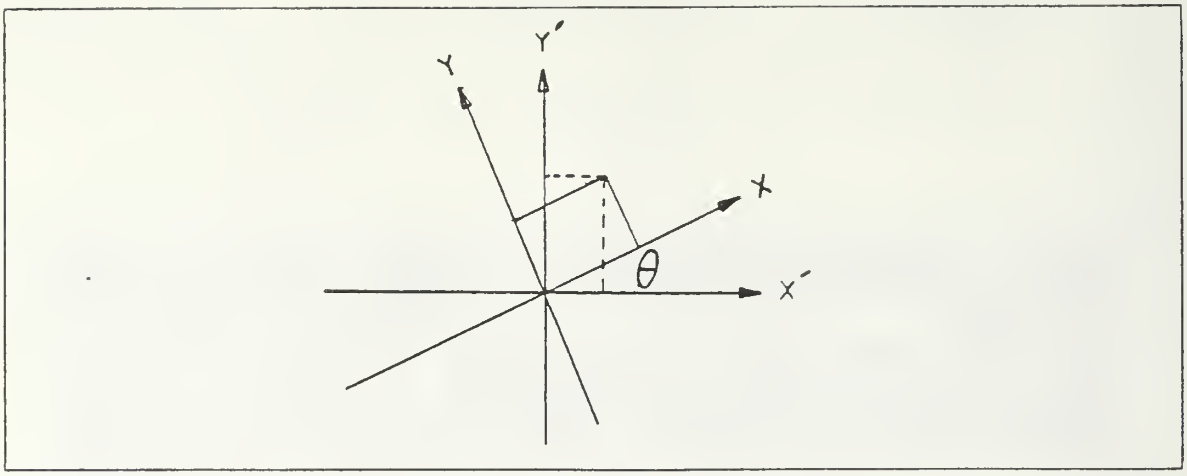


Figure 2.19 Rotation Process.

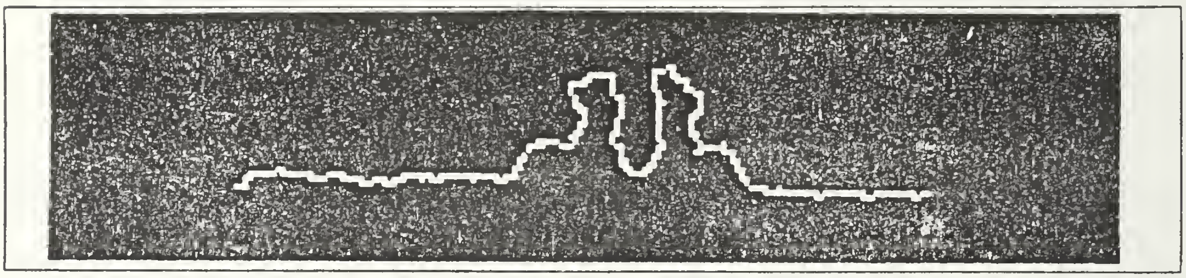


Figure 2.20 Rotated Profile of a CGN at a Range of 45000 feet.

III. FOURIER COEFFICIENT METHOD

We have obtained the ship profile in the previous chapter. To extract features out of this profile for classification purposes, we will use the Fourier Transform method.

The Fourier transform of the ship profile showed that the transform coefficients depend upon the ship's dimensions, its superstructure, and the distance between the camera and the ship. If the profile f(x) is a discrete function with 128 sample points. The discrete Fourier transform can be written as

$$F(u) = \frac{1}{N}\sum_{x=0}^{M-1} f(x)\exp(-j2\pi u x/N) \tag{3.1}$$

For u, x = 0, 1, 2, ..., N-1. N is the total number of samples.

If the direct calculation of the discrete Fourier transform is chosen, the number of complex multiplication and addition will be equal to N ; i.e. to obtain F(0) would require complex multiplication and addition N times. In order to reduce the computation, we use the fast Fourier transform algorithm. Thus, equation 3.1 [Ref. 2] can be separated into F_{even} (u) and F_{odd} (u)

$$F_{even}(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(2x)\, W_M^{ux} \tag{3.2}$$

$$F_{odd}(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)\, W_M^{ux} \tag{3.3}$$

$$W = \exp(-j2\pi/M) \tag{3.4}$$

$$M = \frac{N}{2} \tag{3.5}$$

$$F(u) = \frac{1}{2}\left[F_{even}(u) + F_{odd}(u) W_{2M}^{U}\right] \tag{3.6}$$

$$F(u+M) = \frac{1}{2}\left[F_{even}(u) - F_{odd}(u) W_{2M}^{U}\right] \tag{3.7}$$

Using this method, we have reduced the total number of complex multiplication and addition to Nlog N. In this case, we have N = 128, thus, the total number of complex multiplication and addition will be 896.

First, we divide the rotated profile image into 128 divisions. Since the distance of each division is equal to the amount of the pixel between the bow and the stern of the ship divided by 128, we use the distance perpendicular from the horizontal line between bow and stern to the highest point of the superstructure as the sampled values. The result of the fast Fourier transform is a complex number. Then we use

$$M(u) = \log\left[1 + G(u)\right] \tag{3.8}$$

G(u) is the magnitude of F(u). A value of 1 is add to the magnitude to avoid negative logarithm result, the results obtained are as shown in Table I and

1. Figure 3.1 - Destroyer
2. Figure 3.2 - Container
3. Figure 3.3 - Freighter
4. Figure 3.4 - Replenishment oiler
5. Figure 3.5 - Tank landing ship

6. Figure 3.6 - Frigate
7. Figure 3.7 - Guided missile cruiser
8. Figure 3.8 - Guided missile destroyer

On the results minor difference of the shape of the Fourier coefficients can be noticed visually. But it is difficult to implement a program to detect these minor difference in shape. Therefore, a Second method was used to handle this problem.

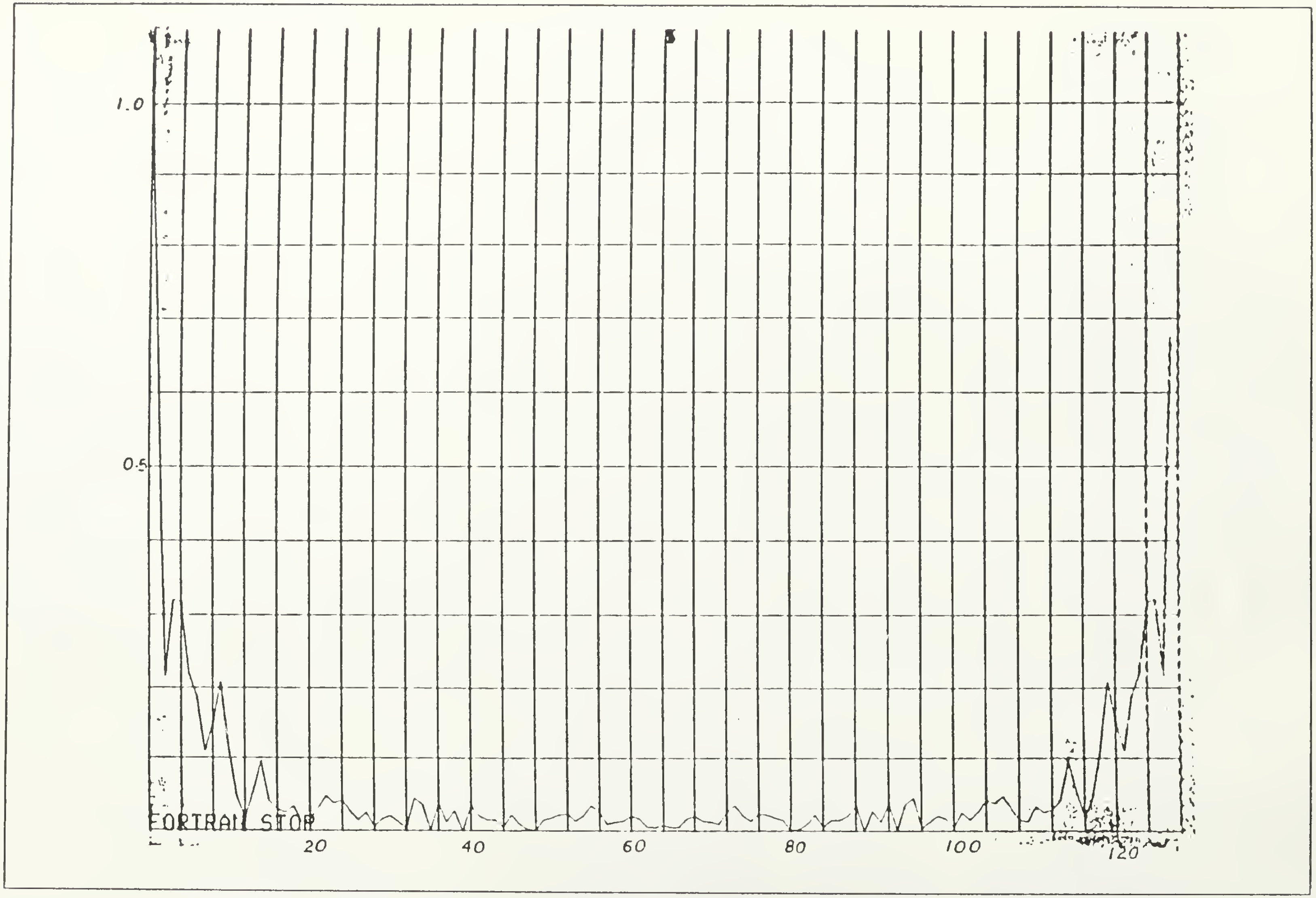


Figure 3.1 Logarithmic Magnitude of the Fourier Transform of a DD.

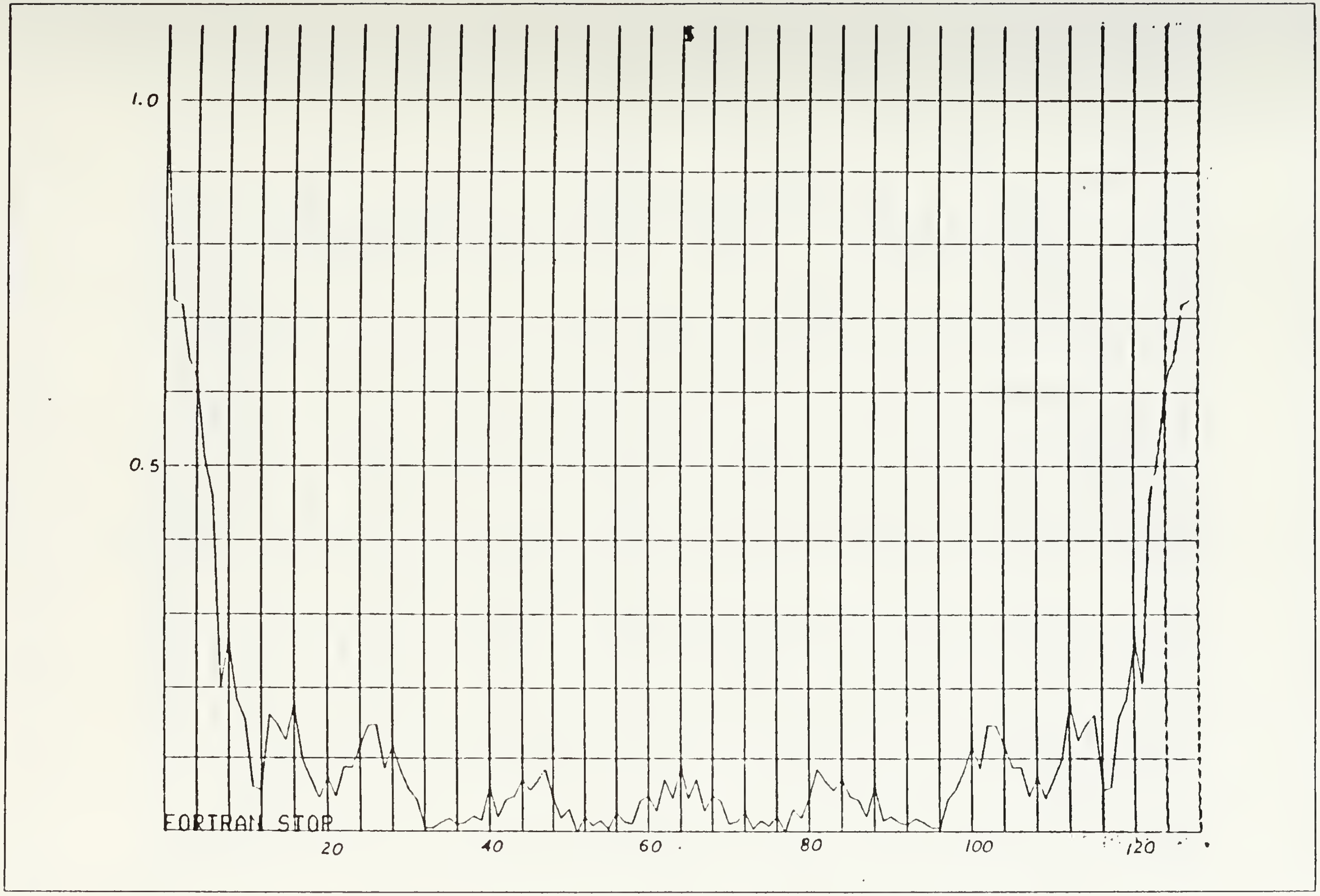


Figure 3.2   Logarithmic Magnitude of the Fourier Transform of a Container.

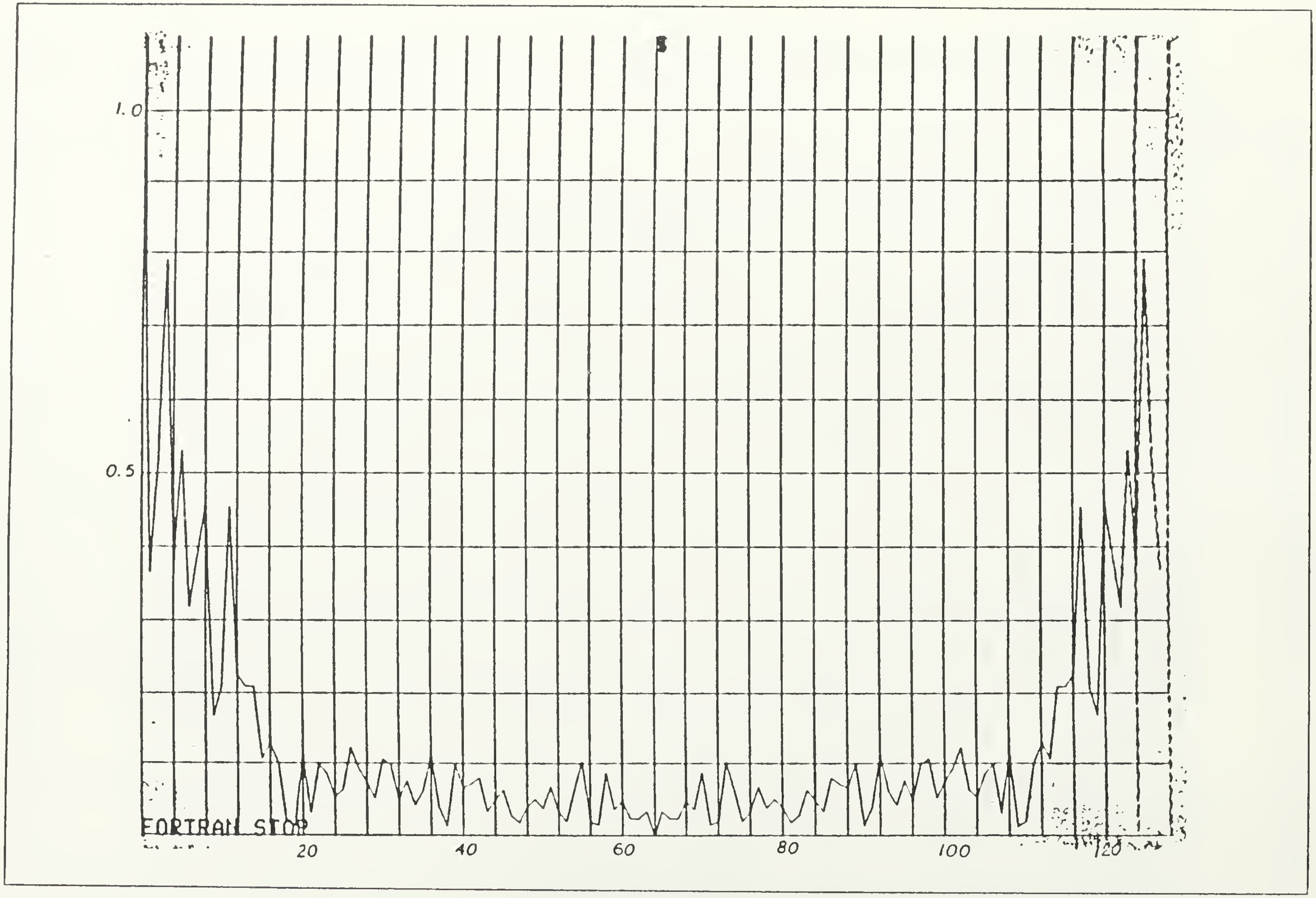


Figure 3.3 Logarithmic Magnitude of the Fourier Transform of a Freighter.

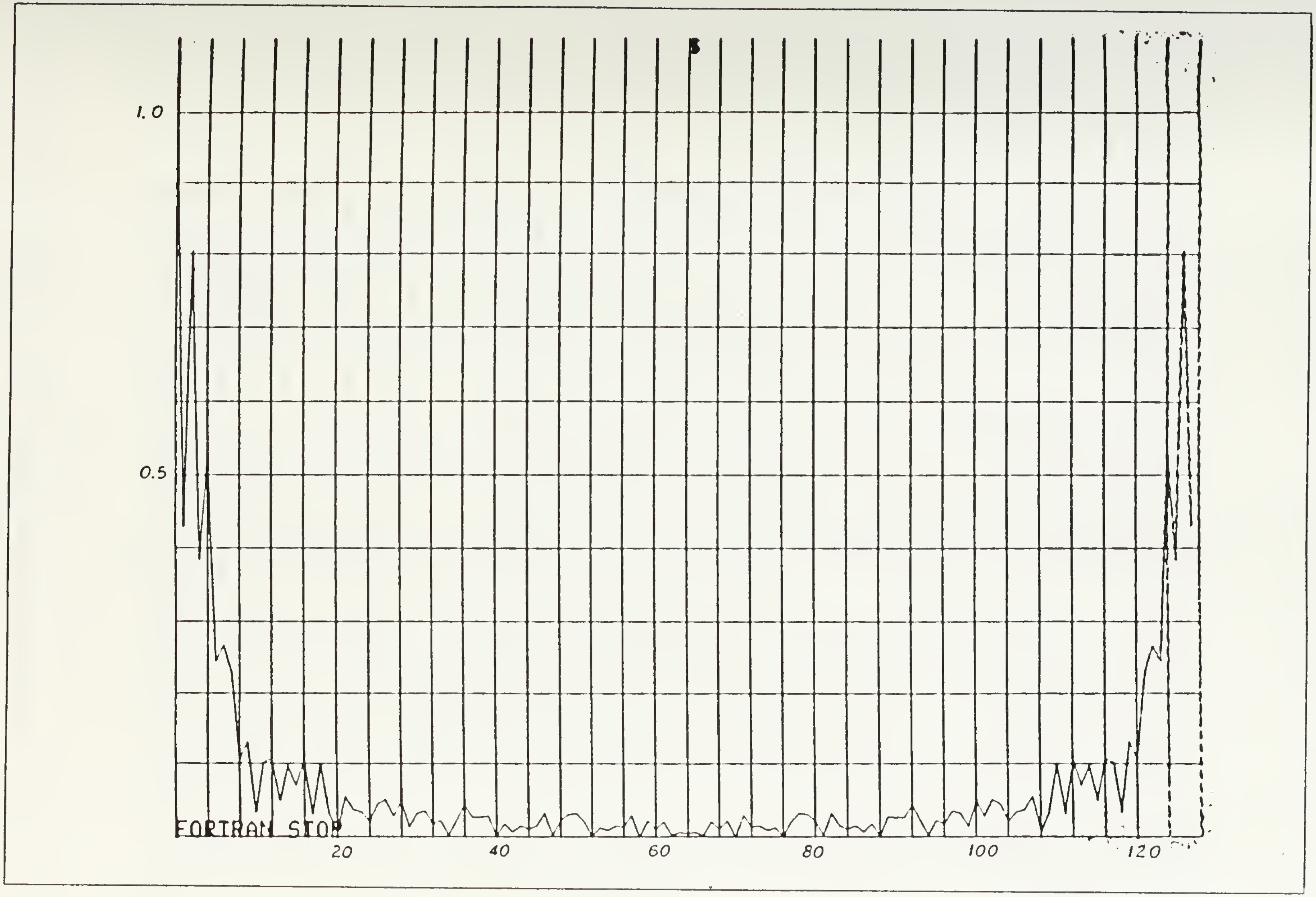


Figure 3.4 Logarithmic Magnitude of the Fourier Transform of a AOR.

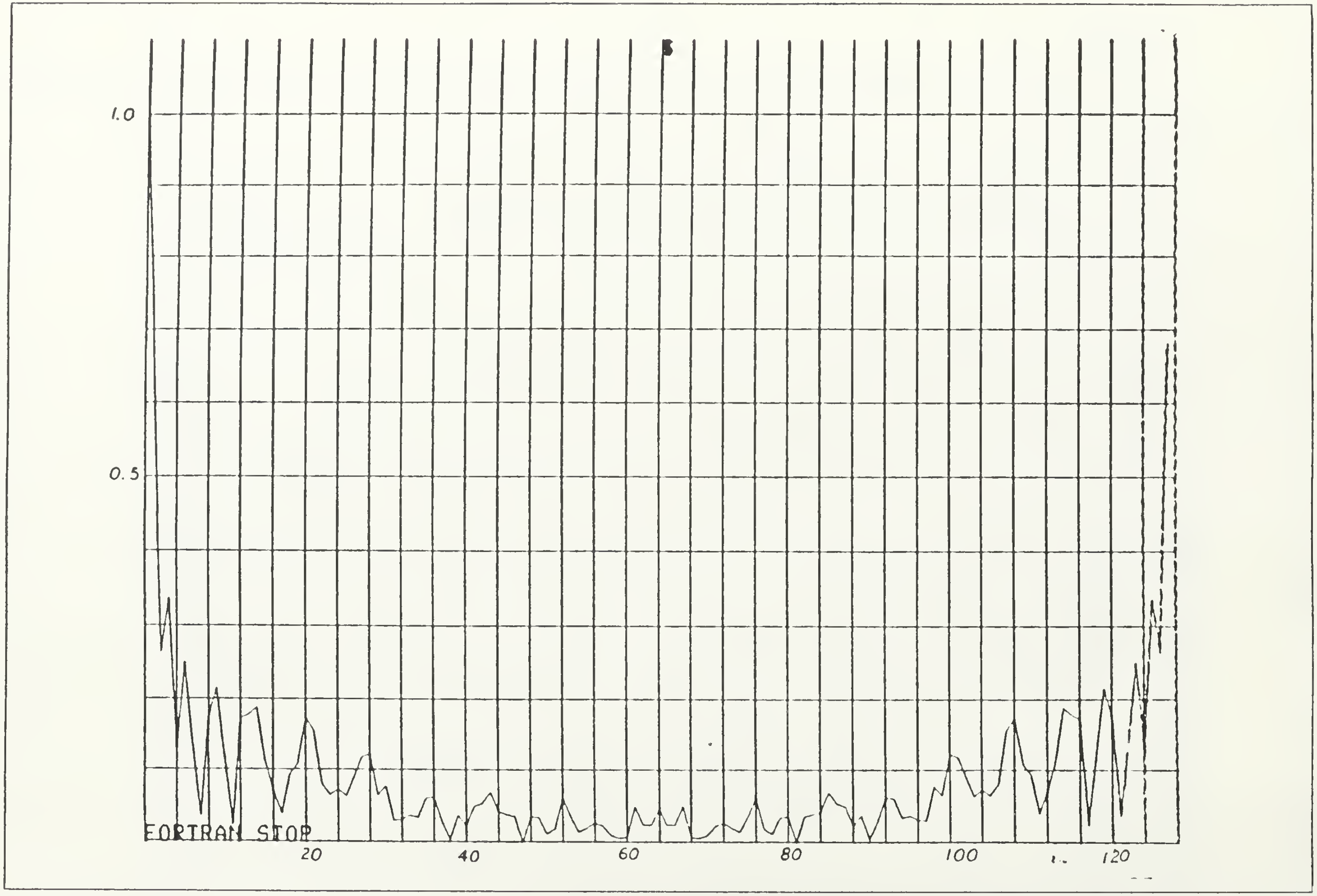


Figure 3.5 Logarithmic Magnitude of the Fourier Transform of a LST.

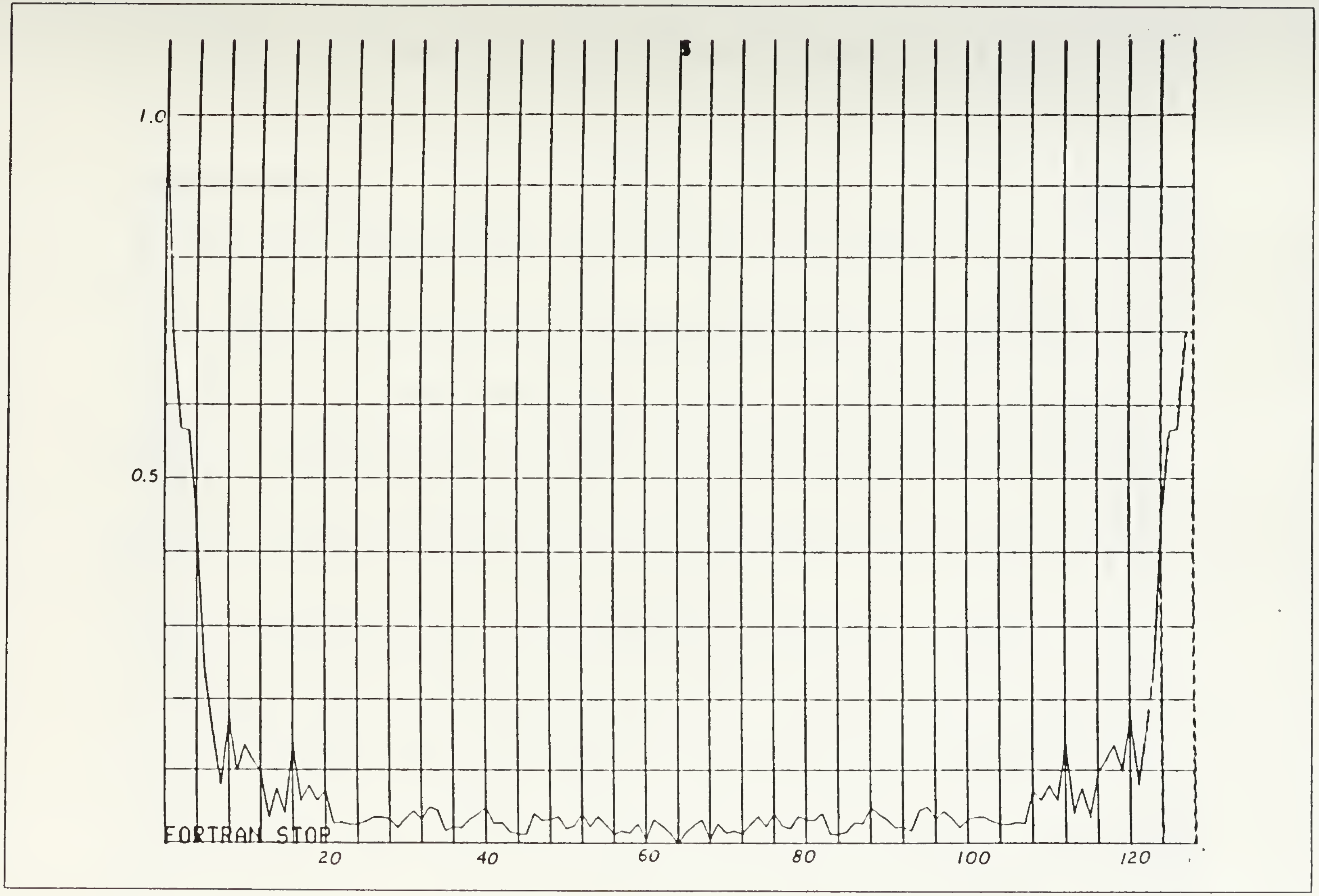


Figure 3.6 Logarithmic Magnitude of the Fourier Transform of a FF.

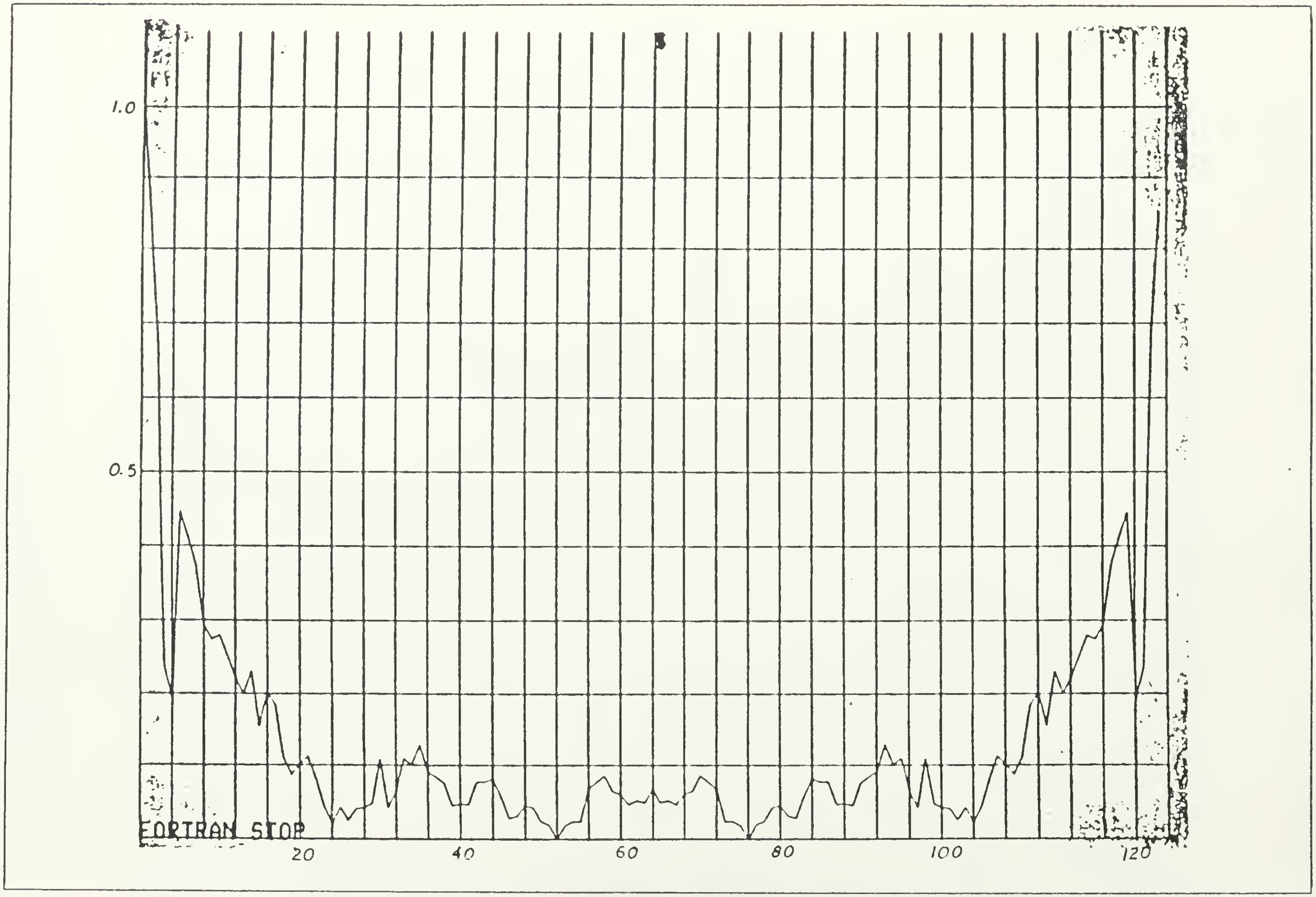


Figure 3.7 Logarithmic Magnitude of the Fourier Transform of a CGN.

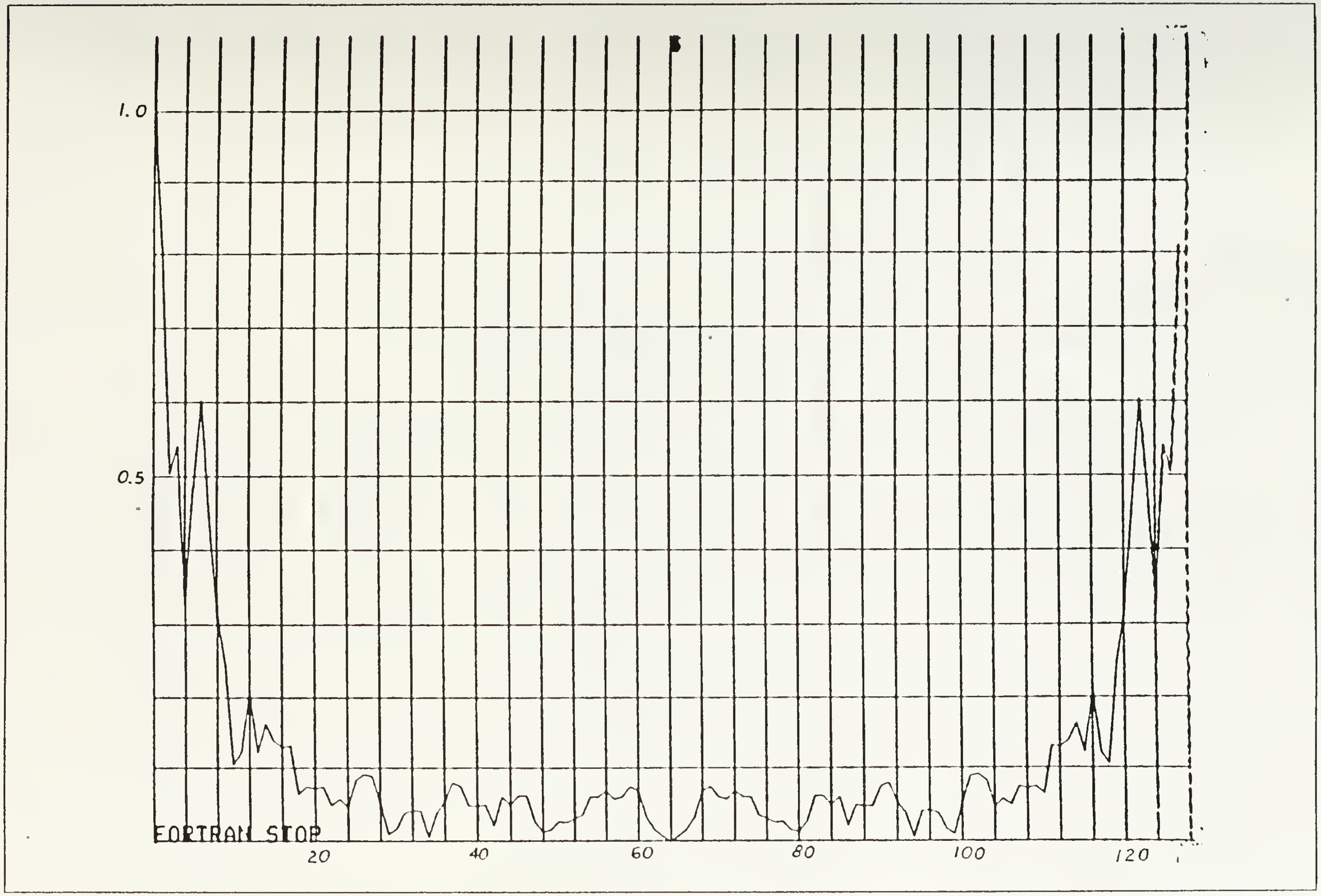


Figure 3.8   Logarithmic Magnitude of the Fourier Transform of a DDG.

TABLE I

Logarithmic Magnitude of Ships

F(N) (1E-1)	DD (1E-1)	Cont. (1E-1)	Frig. (1E-1)	AOR (1E-1)	LST (1E-1)	FF (1E-1)	CGN (1E-1)	DDG (1E-1)
DC	1.E1	1.E1	1.E1	1.E1	1.E1	1.E1	1.E1	1.E1
F(1)	6.74	7.23	3.66	4.28	6.81	6.97	7.78	8.10
F(2)	2.17	7.18	5.17	8.04	2.64	5.67	6.37	5.04
F(3)	3.21	6.45	7.88	3.84	3.35	5.64	2.38	5.40
F(4)	3.17	6.18	3.76	5.28	1.28	4.43	7.6E-1	3.26
F(5)	2.20	5.09	5.31	2.45	2.49	2.40	3.53	4.81
F(6)	1.88	4.57	3.17	2.66	1.36	1.52	3.48	6.02
F(7)	1.10	2.03	3.90	2.28	3.6E-1	8.1E-1	3.44	4.50
F(8)	1.50	2.65	4.58	1.07	1.74	1.76	3.35	3.14
F(9)	2.07	1.83	1.67	1.32	2.14	9.9E-1	3.40	2.46
F(10)	1.13	1.57	2.07	3.39E1	1.19	1.13	3.24	1.05
F(11)	4.6E-1	6.0E-1	4.53	1.03	2.46	1.14	2.62	1.23
F(12)	1.6E-1	5.6E-1	2.23	1.07	1.73	9.8E-1	1.80	2.09
F(13)	5.4E-1	1.62	2.09	5.0E-1	1.78	3.5E-1	8.5E-1	1.23
F(14)	9.6E-1	1.48	2.09	9.8E-1	1.87	7.4E-1	6.7E-1	1.62
F(15)	4.1E-1	1.25	1.07	7.0E-1	1.14	4.1E-1	1.19	1.38
F(16)	2.9E-1	1.77	1.30	1.05	7.1E-1	1.36	1.25	1.30

DD = Destroyer at range 77000 feet.

Cont = Cointainer at range 28000 feet.

Frig= Freighter at range 40000 feet.

AOR = Replenishment oiler at range 78000 feet.

LST = Tank landing ship at range 51000 feet.

FF = Frigate at range 49000 feet.

CGN = Guided missile cruiser at range 45000 feet.

DDG = Guided missile drestroyer at range 41000 feet

A. VARIATION OF SHIP SUPERSTRUCTURE WITH RANGE

One of the practical problem of using the ship profile is that it is sensitive to range variations. Close ship profile has more details than the far away ship profile. The dependency of the geometric size of the profile with the range is discussed in this section.

Assume that the object is centered on the camera axis. The field of view of the camera, the number of the pixel in the image, the size of the image, and the size of the object are known. Our problem is to determine the distance between the camera and the object.

1. Background

The camera system is similar to a human eyes system. The light reflected from the object goes into the eyes. The image of the object falls on the retina, the signal is sent to the brain in some electrical from, and the brain changes it to a from that human can perceive. But, in the camera system film or senser are used to pick up the image. The function of a camera is shown in Figure 3.9

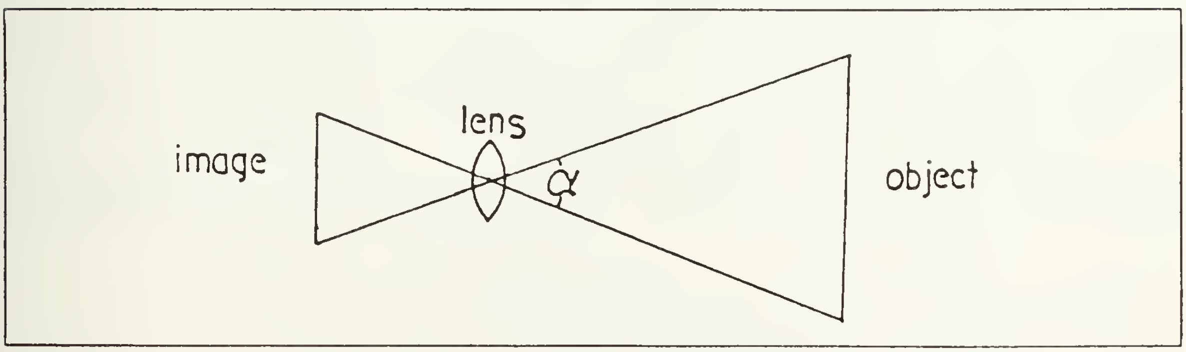


Figure 3.9 The System of the Camera.

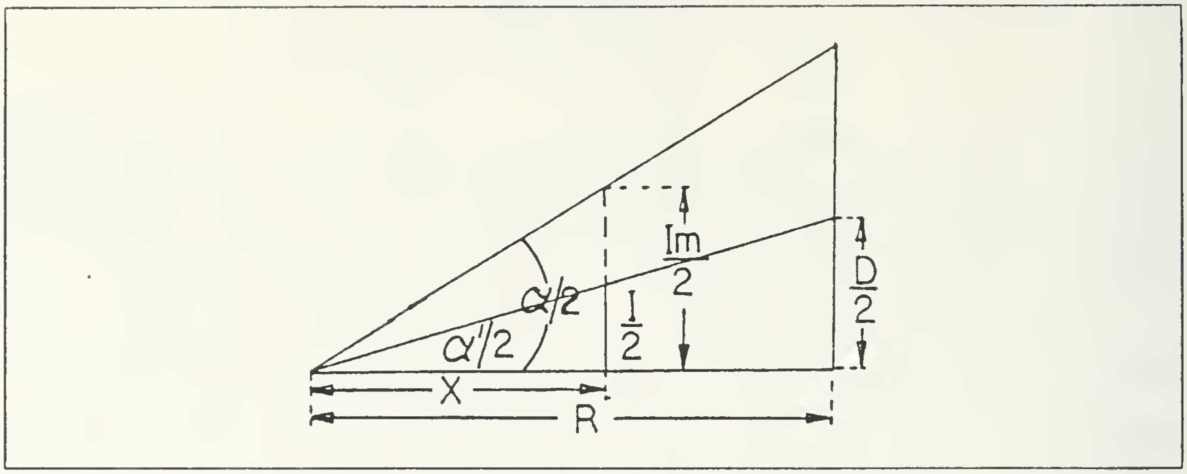


Figure 3.10 Range Calculation.

The distance between the lens and the image plane can be adjusted in order to have a clear image on the film. The camera has a field of view (FOV) angle as shown in Figure 3.9. The object has to be in the field of view of the camera. The determination of the distance is shown in Figure 3.10

For simplicity of calculation the inside of the camera is flipped to the same side as the object as shown in Figure 3.10. when the angle of the FOV is α, I/2 is the half of the full image size, D/2 is the half of the dimension of the object, X is the distance from the lens to the image, and R is the distance from the lens to the object.

Assume that the distance I/2, distance Im/2, and angle α/2 are known. Then, the distance R can be determined by

$$X = \frac{Im}{2} \tan\frac{\alpha}{2} \qquad (3.9)$$

$$\tan\frac{\alpha'}{2} = \frac{I}{2}\frac{1}{X} \qquad (3.10)$$

$$R = \frac{D}{2}\frac{1}{\tan\frac{\alpha}{2}} \tag{3.11}$$

Assume that the angle resolution of the pixel of the field of view of the camera is equal to 0.2E-3 radian per pixel. The number of pixel of the frame is equal to 256. The size of an image is I pixels. The dimension of the object D in feet is known. The field of view in angle is

$$\alpha = (0.2E-3)256 \tag{3.12}$$

$$X = \frac{256}{2}\frac{d}{\tan\frac{\alpha}{2}} \tag{3.13}$$

where d is in unit of pixel.

$$\tan\frac{\alpha'}{2} = \frac{I}{2}\frac{d}{X} \tag{3.14}$$

$$\tan\frac{\alpha'}{2} = \frac{I}{256}\tan\frac{\alpha}{2} \tag{3.15}$$

$$R = \frac{D}{2}\frac{1}{\tan\frac{\alpha'}{2}} \tag{3.16}$$

$$R = \frac{128D}{I\tan 0.0256} \tag{3.17}$$

When the length D of the object is known, from equation 3.17 we can estimate the distance from lens to the object and is shown in Table II

TABLE II

Range Estimation

CLASS	I(pixel) MEASURE	D (ft) KNOWN	R' (kft) CALCULATE	R(kft) RADAR DIS	$\frac{(R-R')100}{R}$
DD1	96	418	21.766	77	71.71
DD2	80	418	26.119	85	69.27
AOR1	107	659	30.787	78	60.53
AOR2	96	659	34.315	85	59.63
LST1	176	522.3	14.834	51	70.91
LST2	134	522.3	19.484	57	65.82
CGN1	147	565	19.213	45	57.31
CGN2	119	565	23.734	55	56.85
DDG1	126	437	17.337	47	63.11
DDG2	90	437	24.247	64	62.08

DD1,DD2 = Destroyer at range 79000 and 83000 feet.

AOR1,AOR2 = Replenishment oiler at range 78000 and 88000 feet.

LST1,LST2 = Tank landing ship at range 51000 and 62000 feet.

CGN1,CGN2 = Guided missile cruiser at range 45000 and 64000 feet.

DDG1,DDG2 = Guided missile destroyer at range 41000 and 64000 feet.

The error distance between the estimated distance and calculated distance in percentage is $((R - R')/R)\ 100$.

2. <u>Remark</u>

Calculated distance error in R may come either from the pixel measurement in an image or from the angular resolution estimation. The distance that is stored in the image label has an error of about 1 to 2 kilo-feet. If the angle

of the field of view is accurate, we can estimate the number of pixel in an object image. Then we can classify the object by comparing the number of pixel of the original image with that of the test image. Some known system parameters can help to determine the range of the target ship. The problem is that existing errors in the system parameter causes in the larger errors in the estimated range. Consequently, they are not very useful in classifying the ships.

IV. B-SPLINE COEFFICIENT METHOD

Using B-spline coefficient is another method to describe a ship profile. This method uses the B-spline coefficients to determine the beginning, peak, and area of lumps which contain significant information about the type of the ships. The comparisons of the knot position (in parametric value) from the midships to the peak or beginning of the lump, can be very helpful. Different ships will have different lump positions and sizes.

A. BACKGROUND

A spline function is a piecewise polynomial used to interpolate points. This kind of curve is smooth and the discontinuities in its k-th derivative is as small as possible. In this case, Cubic spline was used, where the first and second derivatives for any set of interpolating points are continuous, while the third derivative may be discontinuous. The reason for using Cubic spline is to keep the third derivative discontinuity as small as possible and the curve as smooth as possible. The B-spline calculation procedure is also very stable. In our case the order of spline used is 3, while the 1-st and 2-nd derivative are continuous. The choice of 4-th order spline function is due to the fact that it is generally sufficient for most ship profile curves. Discontinuity, in a sense, may be stated as the jumps of the third derivative, which is the means to control smoothness of the connecting pieces.

B. B-SPLINE APPROXIMATION WITH FREE KNOT

As mention earlier, the B-spline function is used in the spline approximation with free knot. The free knots is sometimes called uneven or irregular knots; that is no fixed number of knots are used. Furthermore, the spline position need not be on the original curve so as to minimize the number of knot position while preserving most of the details of the original curve.

First, minimize the value of the lack of the smoothness $\eta(\bar{c})$ defined as [Ref. 3]

$$\eta(\bar{c}) = \sum_{j=1}^{n} \left(\sum_{i=-k}^{n} a_{ij} c_i \right)^2 \tag{4.1}$$

where Ci is a coefficient of spline at the knot position. a_{ij} is defined as follows

$$a_{ij} = M_{i,k+1}^{(k)}(t_j+0) - M_{i,k+1}^{(k)}(t_j-0) \tag{4.2}$$

where $M_{i,k+1}$ is the normalize B-spline function and defined as

$$M_{i,k+1}(x) = (t_{i+k+1} - t_i)\Delta_t^{k+1}(t_i, \ldots, t_{i+k+1})\, g_k(t;x) \tag{4.3}$$

and $G_k(t;x)$, t are defined as follows

$$g_k(t;x) \begin{cases} = (t-x)_+^k = (t-x)^k & \text{if} \quad t \geq x \\ = 0 & \text{if} \quad t < x \end{cases} \tag{4.4}$$

$\Delta_f^l(Z_i, Z_{i+l}, \ldots, Z_{i+l})f(t)$ stands for the l-th divided difference of the function f(t) on the point $Z_i, \ldots, Z_{i+l}$ where t is the position values of knots in term of Z parameter. Z parameter is defined as follows

$$Z(I) = Z(I-1) + \left[(X(I) - X(I-1))^2 + (Y(I) - Y(I-1))^2\right]^{1/2} \quad (4.5)$$

$$Z(0) = 0 \quad (4.6)$$

where I is the number of the sampling point and I=1,2,3,...,m.

Second, the smoothing is subjected to a constraint

$$\delta(c) \leqslant s \quad (4.7)$$

where S is the smoothing factor, $\delta(\bar{c})$ is the weighted sum of the square residuals defined as

$$\delta(c) = \sum_{j=1}^{m} w_j \left[Y_j - \sum_{i=-k}^{n} c_i M_{i,k+1}(X_j)\right]^2 \quad (4.8)$$

X_j, Y_j are the values of X and Y at Z parameter of the sampling points; W_j is a weighting factor for all sampling points [Ref. 3] defined as

$$w_j = (\delta Y_j)^{-2} \quad (4.9)$$

where W_j is an estimate of the standard deviation of Y_j. Then, the value of S is in the range $m+\sqrt{2m}$. If nothing is known about the statistical standard deviation of Y_j, each

W can be equal to one and the S can be determined by the trial and error method.

1. Interpolation and Approximation Using B-spline Function

There is a difference between the interpolation and the approximation. In interpolation, the number of knot positions required are equal to the number of sampling points and the value of S in eq(4.7) is small. In this case, the computation time required will increase tremendously. Whereas, in approximation, it is not of great concern that the approximated value at a position be the same value of the sampling point, and most of the information still remain in the original curve. Thus, decreasing the value of S will result in the large number of the knot positions and the final curve obtained will be similar to the original curve.

The justification for using approximation rather than interpolation approach is that though the resulting curve may not be the best fitting curve, it is smooth and close enough to the original. The approximation spline function has less number of knots than the number of samples, which reduces the total processing time. In approximation approach, when the appropriate choice of S value is made, it will, in turn, generate sufficient number of knot positions required to provide a close approximation to the original curve. The ratio of sampling point to spline coefficient is 10 to 1 as shown in Figure 4.1. In Figure 4.1, a guided missile cruiser ship at a distance of 45000 feet, with 290 sampling points and the associated B-spline knots are shown. The original samples are plotted in solid curve. The knot position are show by small circles in Figure 4.1. After B-spline approximation, the number of the knots are reduced to 33. Hence, this technique is essentially a kind of data compression technique.

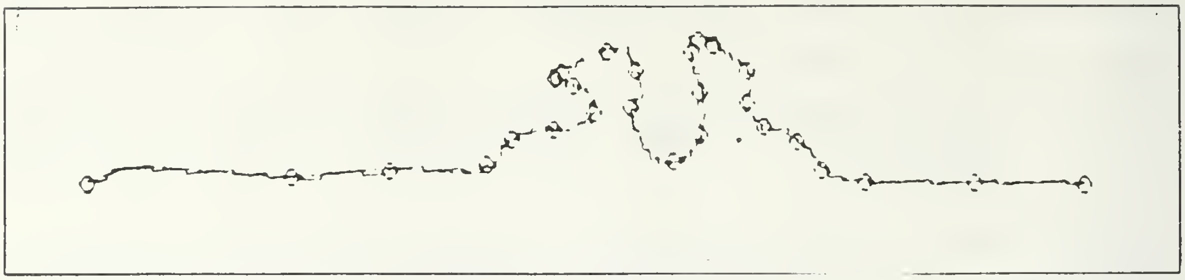


Figure 4.1 Plot of the Original Data and the Approximate with Free Knot.

C. TO DETERMINE THE KNOT POSITION AND THE B-SPLINE COEFFICIENTS

The approximate smoothing factor in eq(4.7) is to be calculated before using the subroutine "PARAM" [Appendix C]. In the subroutine, there is a check on S factor every time it is run. If the S input values do not satisfy or meet the criteria, the program will return with a error code IER.

There is a parameter NEST which is set to a constant value. This value determine the dimension of the knot positions which relates to the array T(NEST), Cx(1..NEST) and Cy(1..NEST). The NEST is an overestimate of the dimension of the arrays set by the user. The limitation on the value of NEST for subroutine "PARAM" are as follows [Appendix C]

1. 2k+2 NEST M+k+1
2. Typically, value of NEST M/2

where M is the number of the total sampling points and k=3 is the highest order in the B-spline function.

The subroutine "PARAM" produces several outputs as, N the total number of knot positions, T(N) the array of the value of knot positions in Z parameter, Cx(N) and Cy(N), B-spline coefficient of X functions and Y functions for knot positions defined in array T(N).

1. Limitation on B-spline Coefficient Determination

If the value of NEST is too small, the user will receive error code IER and the number of knot positions will appear to be dense in the first part of the curve. While sparse in the other part. The example of an incorrect selection of NEST is shown in Figure 4.2.

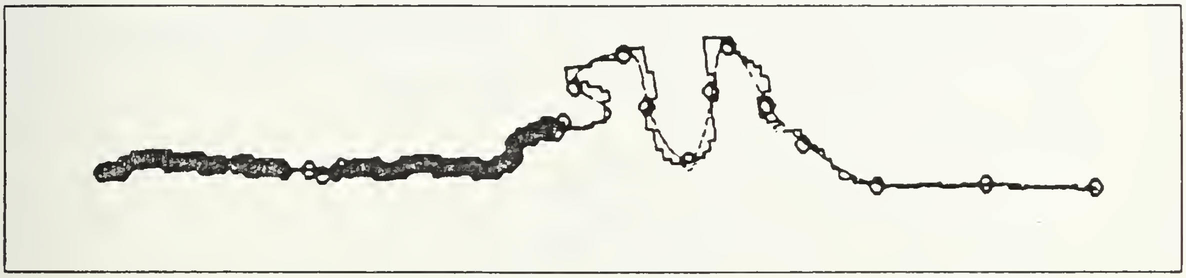


Figure 4.2 Knot Selection if the NEST Parameter is too Small.

Another problem which causes difficulty in programming is that the main program is in PASCAL while the subroutine is in FORTRAN. As for the PARAM program in FORTRAN, the array index value starts from 1, while , for the user PASCAL program it starts from 0. Therefore, in linking the main program to the subroutine, one has to keep in mind of the difference. In addition, the FORTRAN programming logic structure is so complicated that, when an error occurs, it is very difficult to debug and locate the error.

The values of Cx and Cy depend upon uneven knot positions, and they contribute controlling effect to the reconstructed curve which would be both smooth and close to the original curve. In running the B-spline approximation program for the first time, the values of Cx and Cy of the last 4 knots at the right end are zeros. However, in running it again, increase the value of S did not yield zero values of Cx and Cy at those points. This, nevertheless, has no effect on the reconstruction of the curve.

Plots of the reconstructed profile of three ships of the CGN class using different value of S, result in different number of knot positions and are shown for comparison in Figure 4.3, 4.4 and Figure 4.5. The original samples are plotted in solid curve, the reconstructed curve is plotted in dashed line in Figure 4.3.

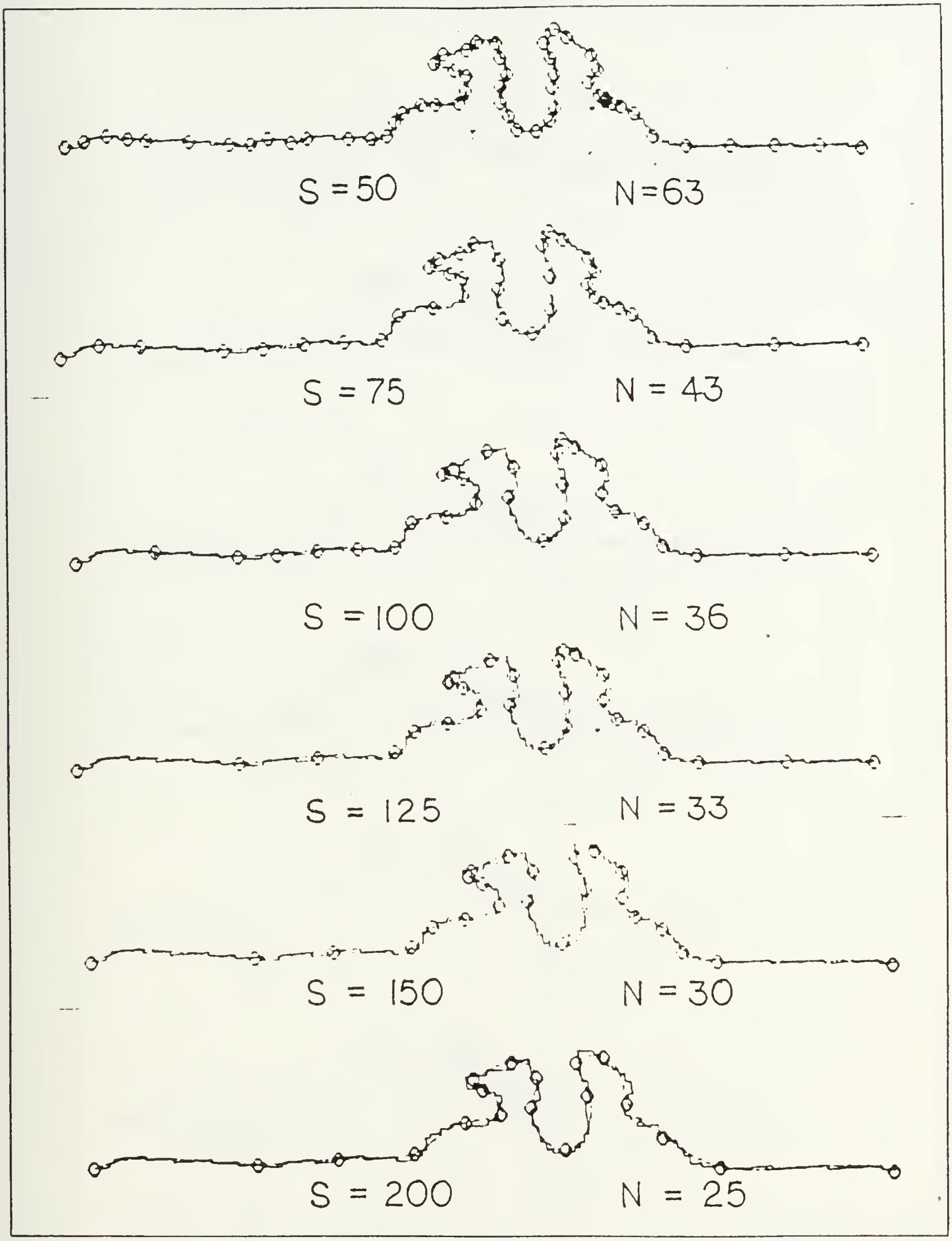


Figure 4.3 A CGN at Range of 45000 feet.

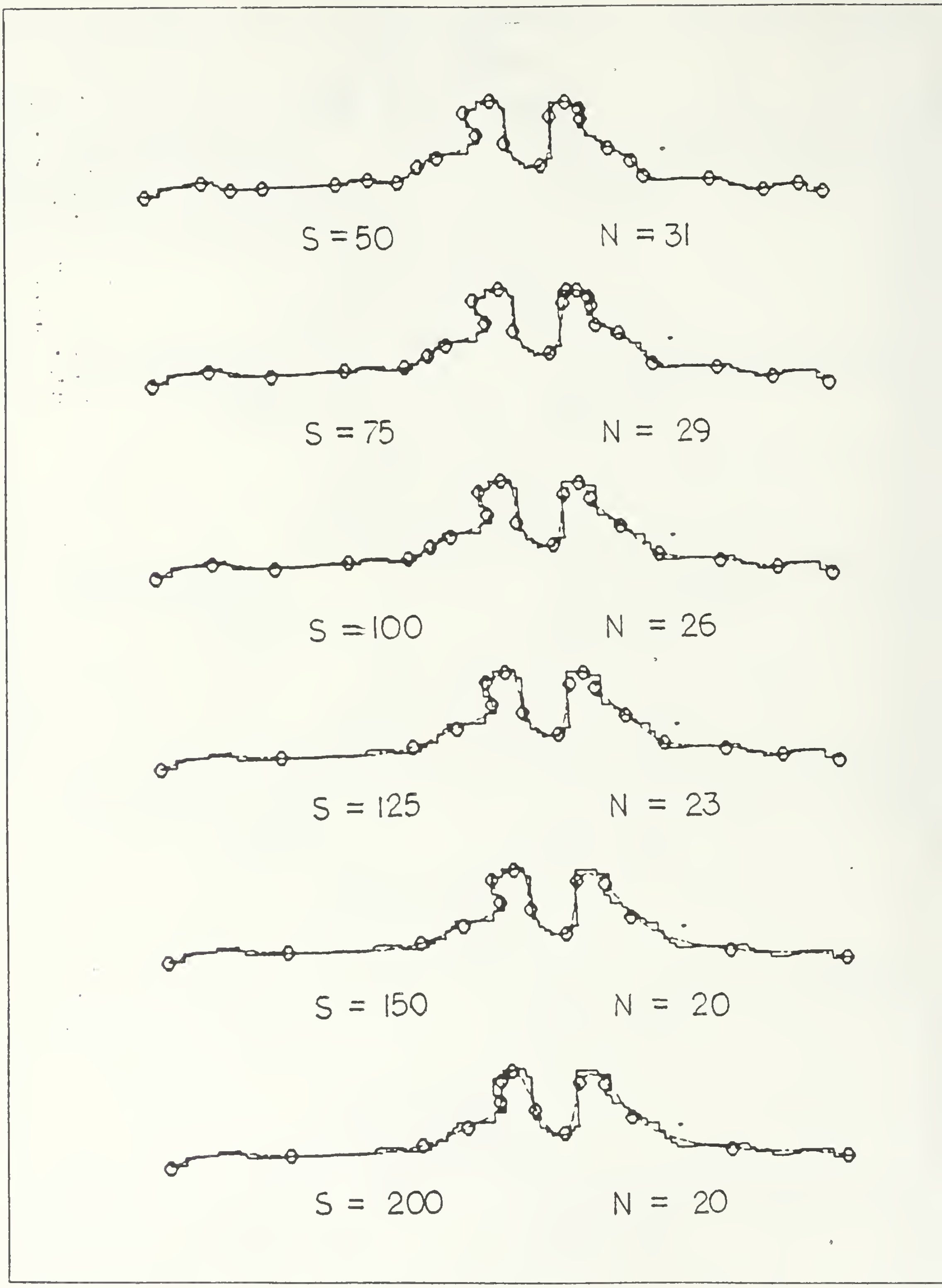


Figure 4.4 A CGN at Range of 55000 feet.

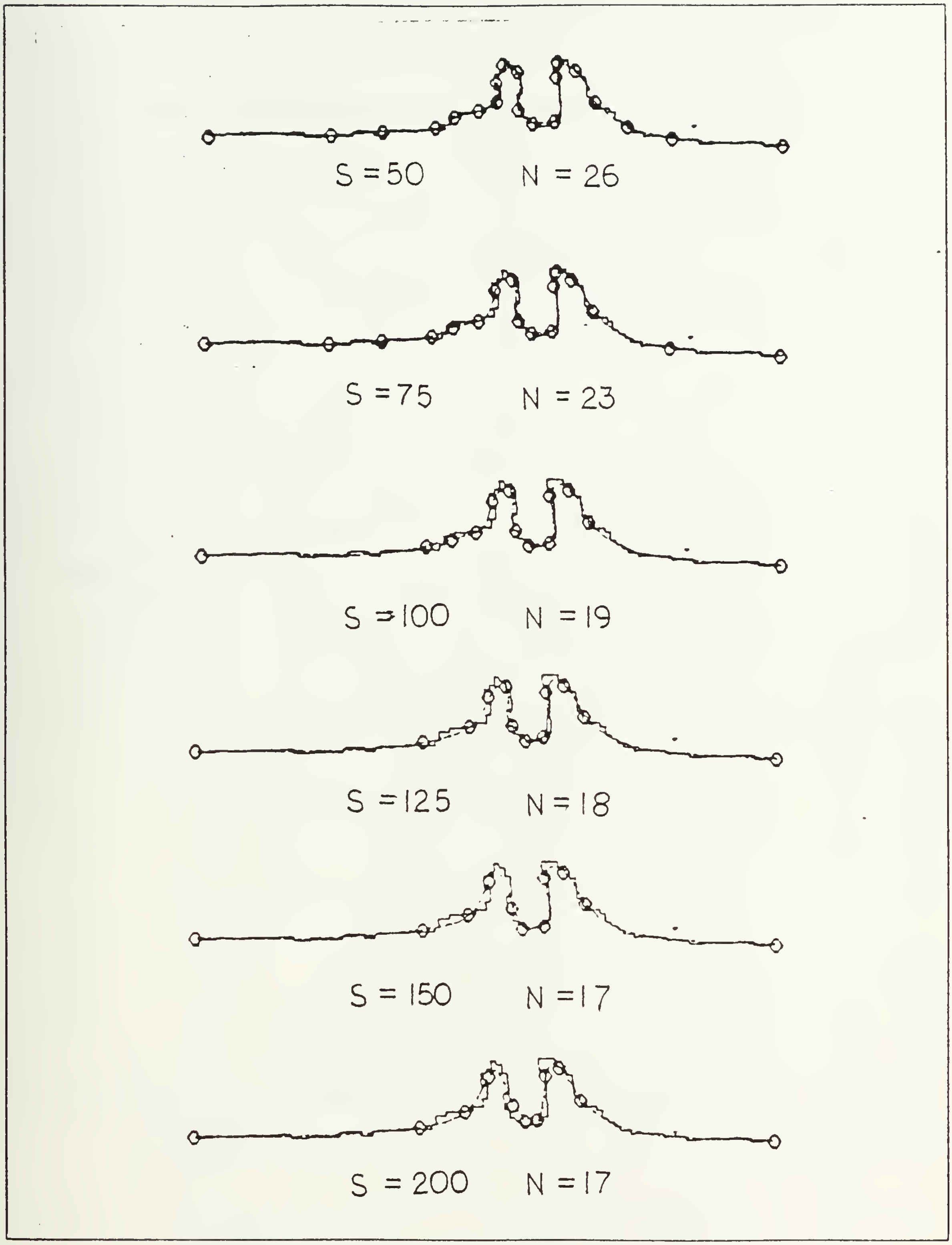


Figure 4.5 A CGN at Range of 64000 feet.

2. Selection of Smoothing Factor Value(S)

It is found by trials and errors using the computer program PARAM that in order to retain maximum information of the profile curve, the number of knot positions required should be in the range of 25 to 35. The number of knot positions depends upon the value of S which must be set accordingly. The appropriate choice of S to satisfy the condition stated above, is important. For the class of a guided missile cruiser(CGN), three ship images at 3 different ranges were selected. Then, run the appropriate program for various S factors to see how the number of knot(N) will vary. The results are shown in Figure 4.6. Plots of N vs S in Figure 4.7 through Figure 4.13 show that, in most cases, the value of N decrease quite rapidly when the value of S is in the range of 0 to 100, and gradually for S factor in the range of 100 to 200, thereafter, the value of N decreases very little. Obviously, the curve seems to decay exponentially. Furthermore, for some classes of ships changes are more pronounced than the others which is probably due to the actual number of knots present in the profile. For guided missile cruiser ship(CGN) with 2 lumps, the number of knot positions required can be 33 as shown in Figure 4.6 We select the factor S to be about 100.

The selection of the factor S depends upon the number of the original sampling points. If the factor S is small, large number of knots are needed. When the factor S is large, small number of knots are needed. When the number of knot and the Cx and Cy coefficients are small, the B-spline coefficients Cx and Cy obtained can not be used to reconstruct the curve close to the original profile.

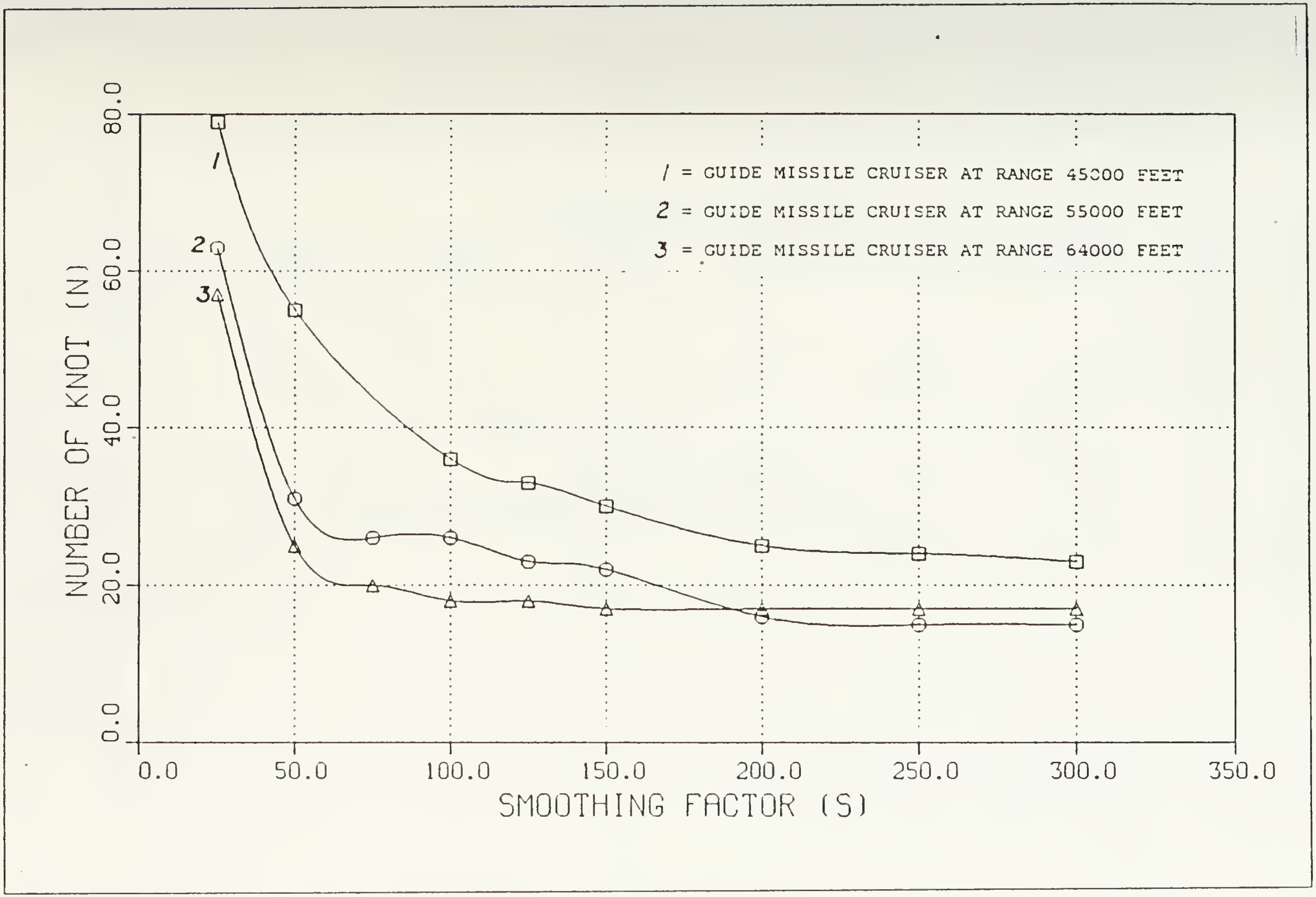


Figure 4.6 Plot N vs S for a CGN.

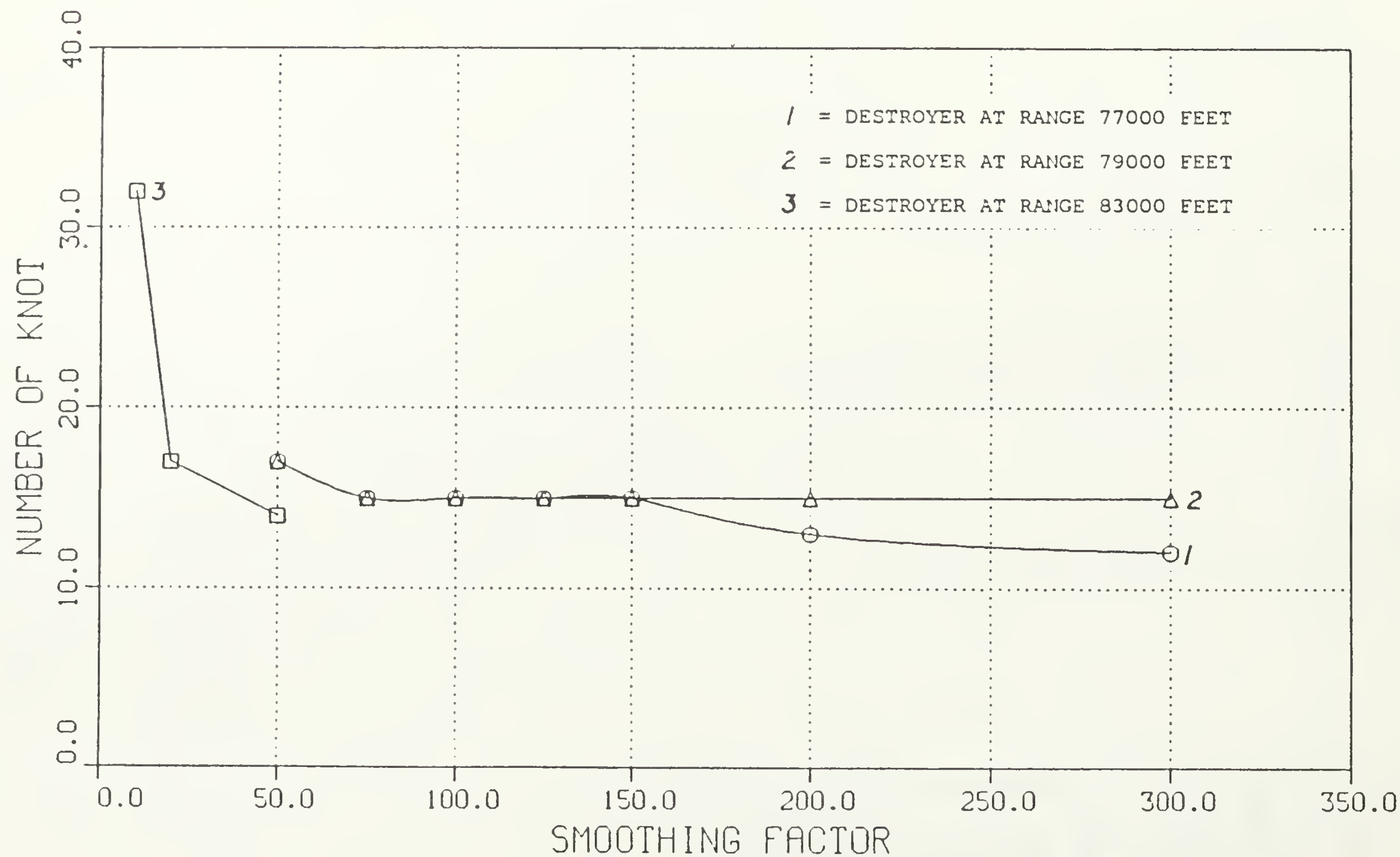


Figure 4.7 Plot N vs S for a DD.

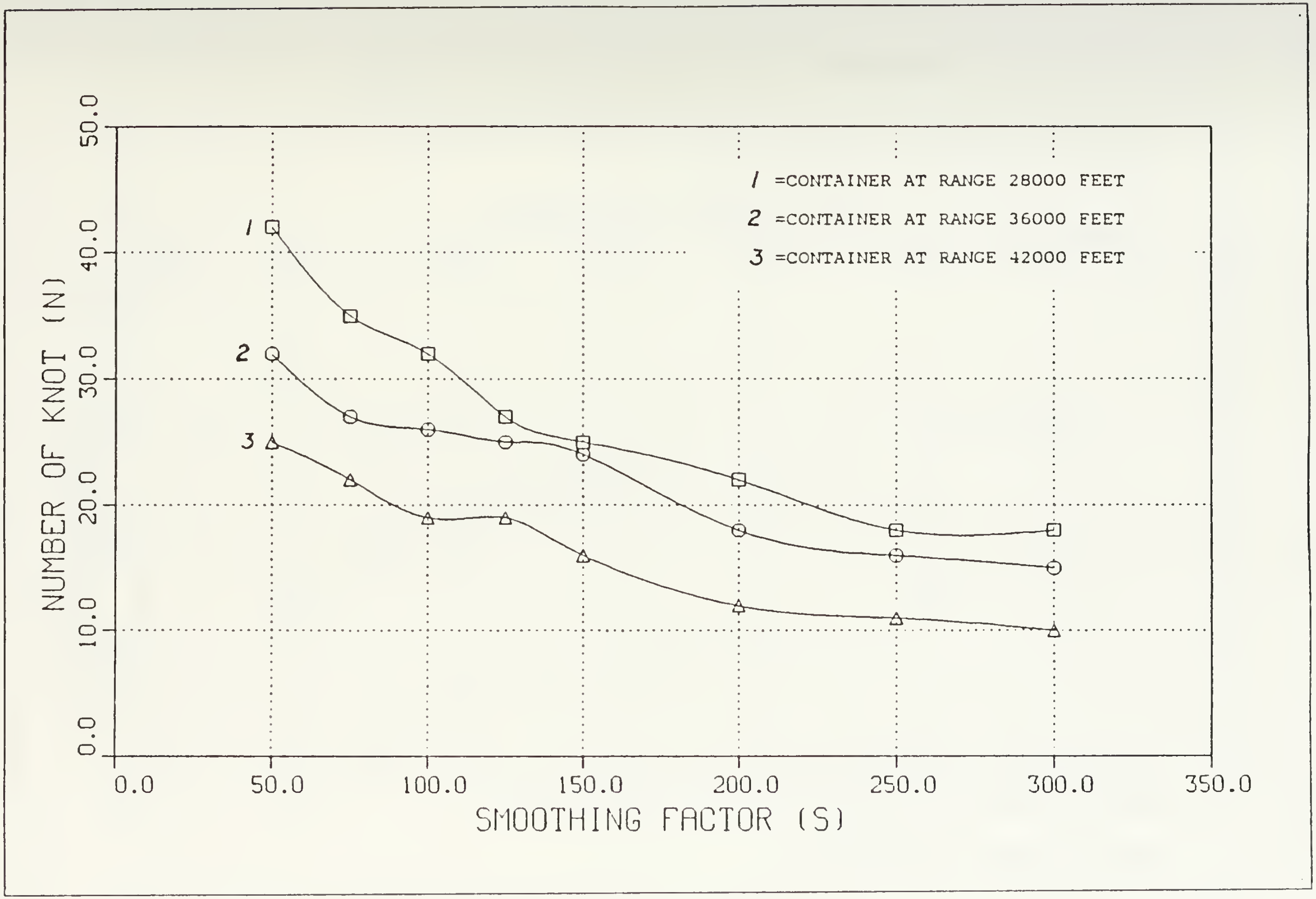


Figure 4.8 Plot N vs S for a Container.

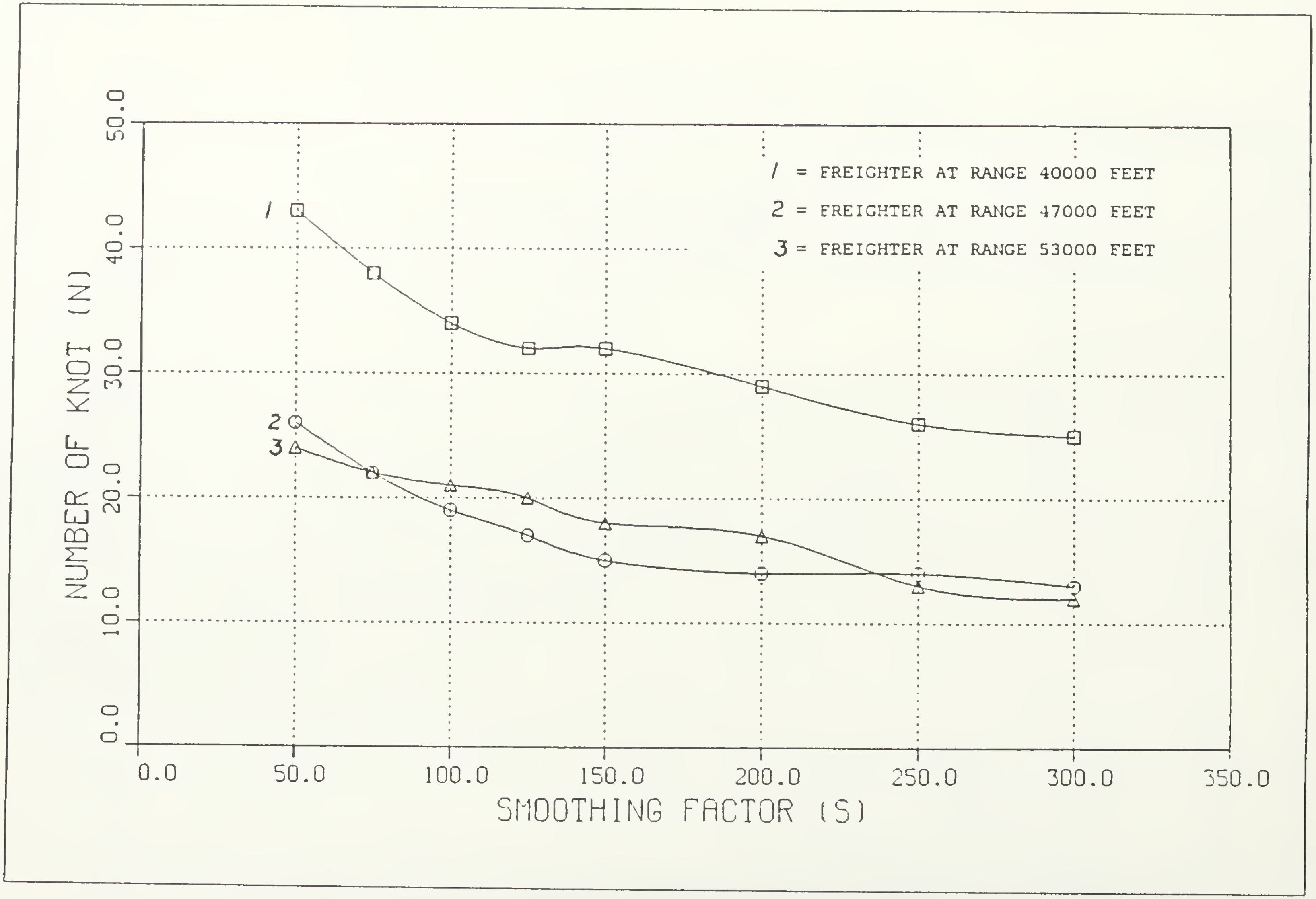


Figure 4.9 Plot N vs S for a Freighter.

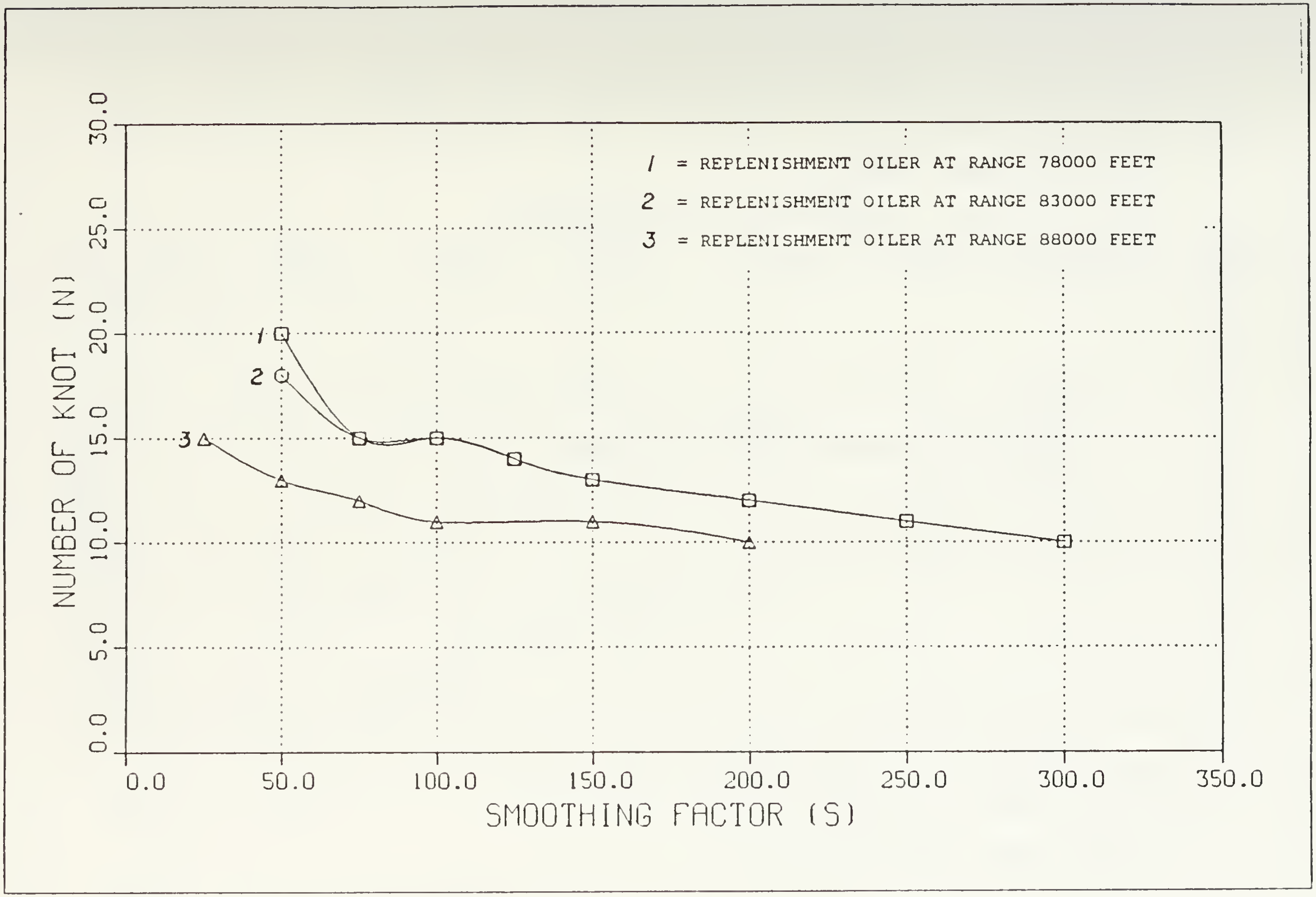


Figure 4.10 Plot N vs S for a AOR.

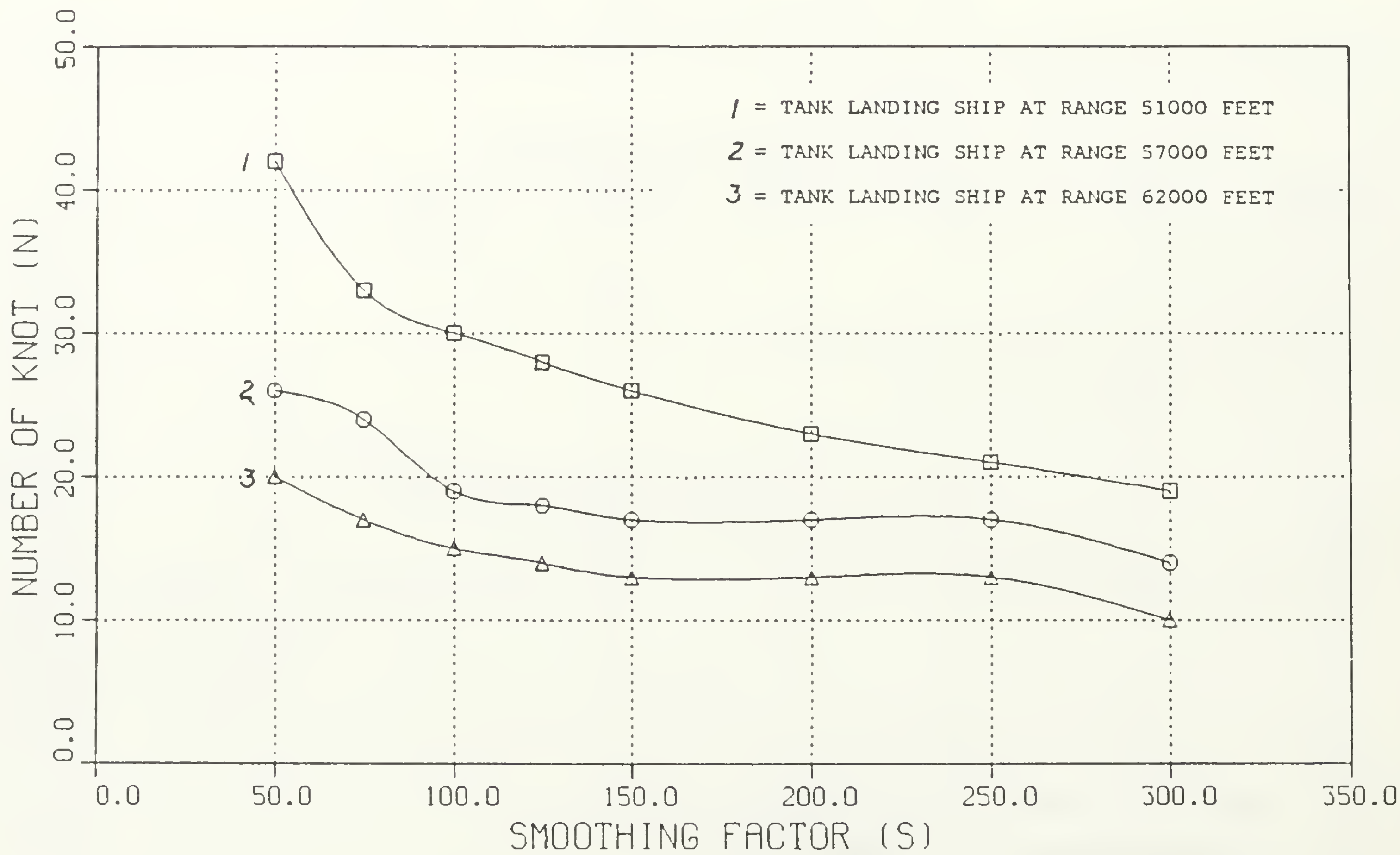


Figure 4.11 Plot N vs S for a LST.

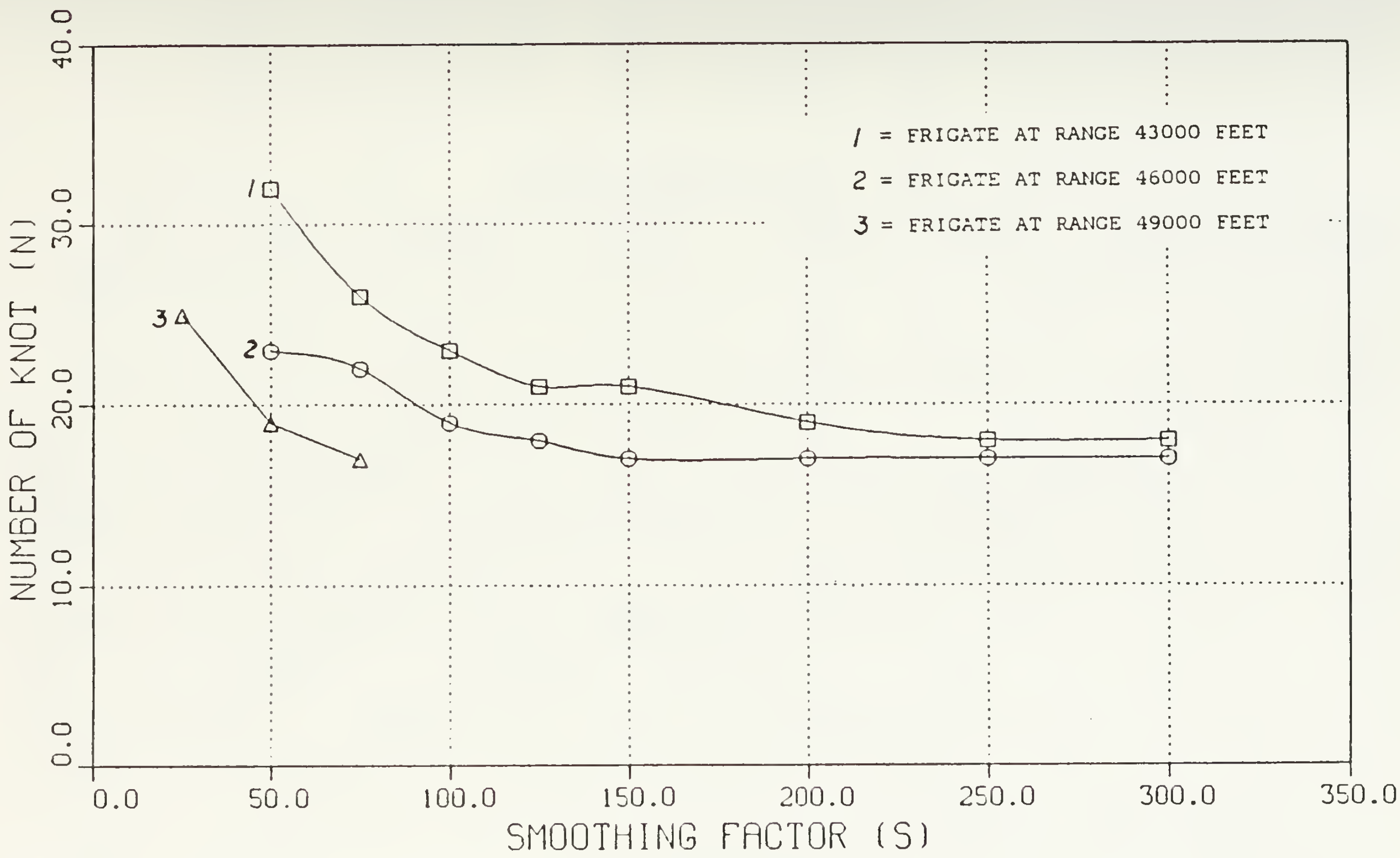


Figure 4.12 Plot N vs S for a FF.

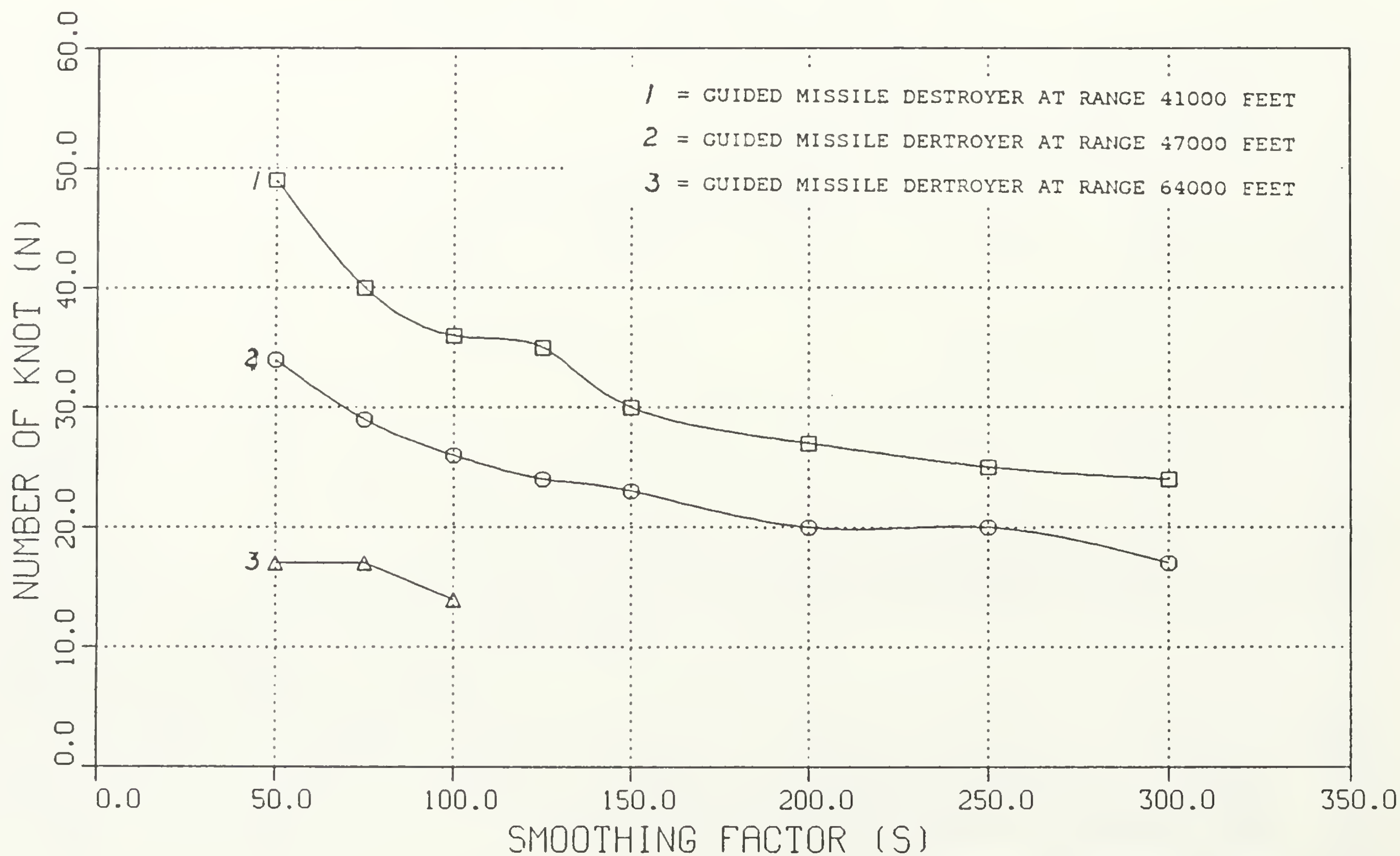


Figure 4.13 Plot N vs S for a DDG.

3. The Output Cx,Cy and Knots Profiles

From the previous study the factor S is selected for each of the class of ship as follows

1. Destroyer(DD), S = 20.0
2. Container, S = 100.0
3. Freighter, S = 100.0
4. Replenishment oiler, S = 20.0
5. Tank landing ship(LST), S = 25.0
6. Frigate(FF), S = 100.0
7. Guided missile cruiser(CGN), S = 125.0
8. Guided missile destroyer(DDG), S = 125.0

The plot of the B-spline coefficients,Cx and Cy,and X,Y at the positions of the sampling points vs the Z parameter are shown in Figure 4.14 through Figure 4.21. Observation and comparisons of the curves show that the values of Cx and Cy exhibit changes similar to that of X and Y except that the variation of values leads that of the X and Y. This is due to the fact that Cx and Cy have to act as the controlling factor for the reconstructed B-spline curve to get the result close to the original curve.

Examination of the plots of the results of X and Y show that when X is increasing monotonically, Y is almost constant; but when X is almost constant, Y is increases or decreases. This behavior relating to profile reconstruction may be explained as follows. For a ship profile when X is increasing and Y not increase too much, this may be interpreted as an almost leveled profile. When X is almost constant, and Y may be increasing or decreasing, it may be interpreted as the beginning or the ending of the lump.

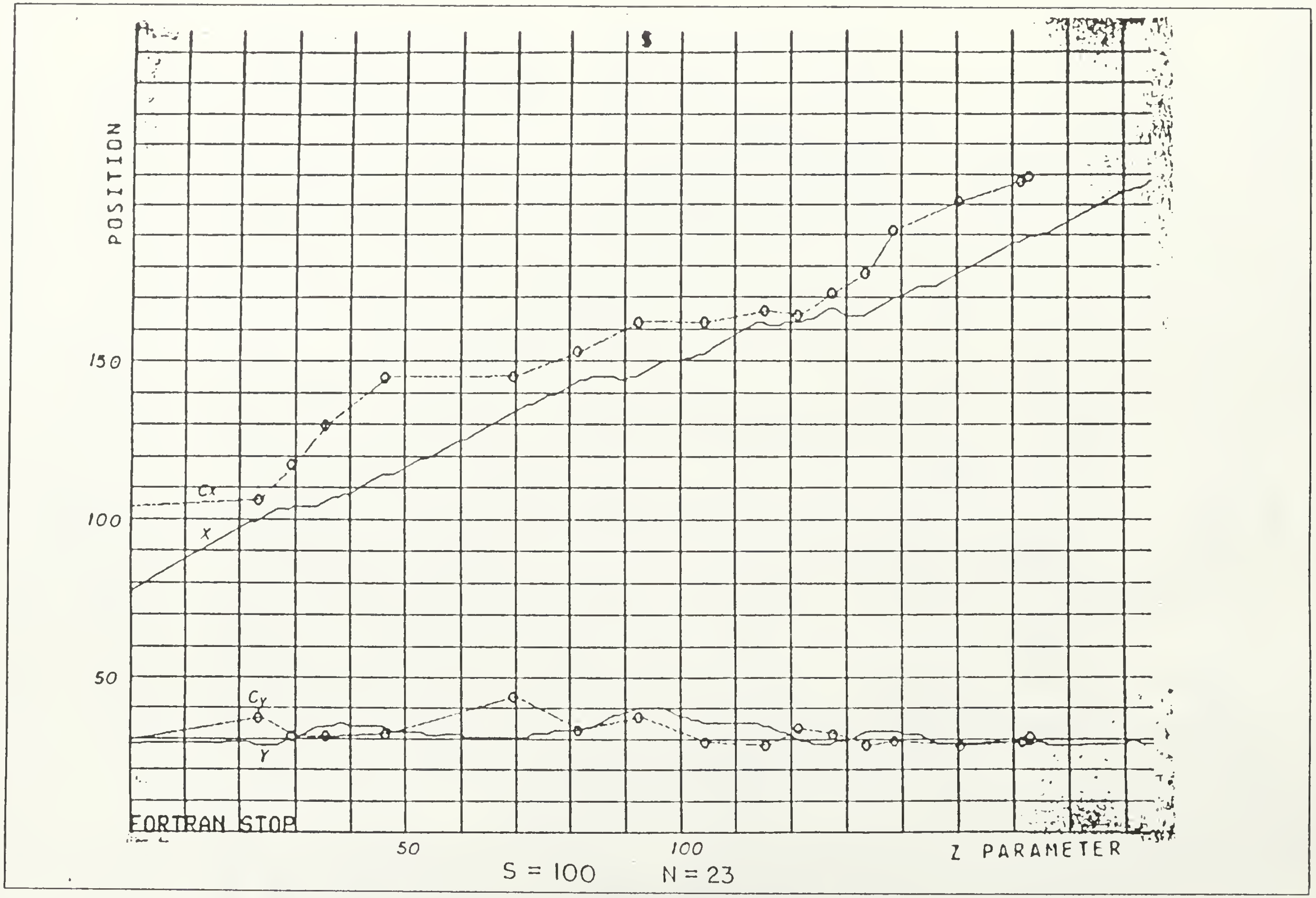


Figure 4.14 Plot X,Y,Cx, and Cy vs Z for a FF at a Range of 43 K-ft.

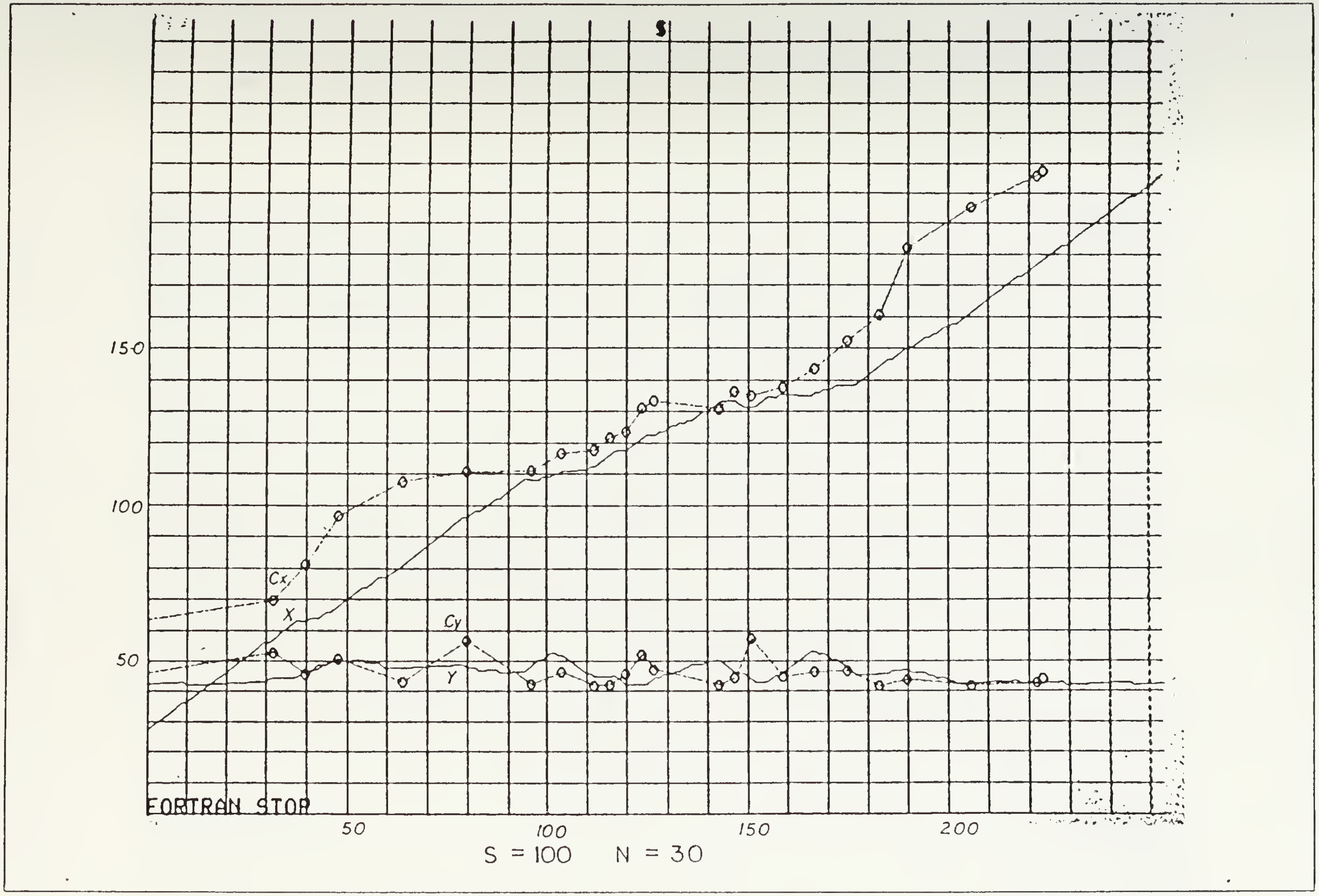


Figure 4.15 Plot X,Y,Cx, and Cy vs Z for a LST at a Range of 51 K-ft.

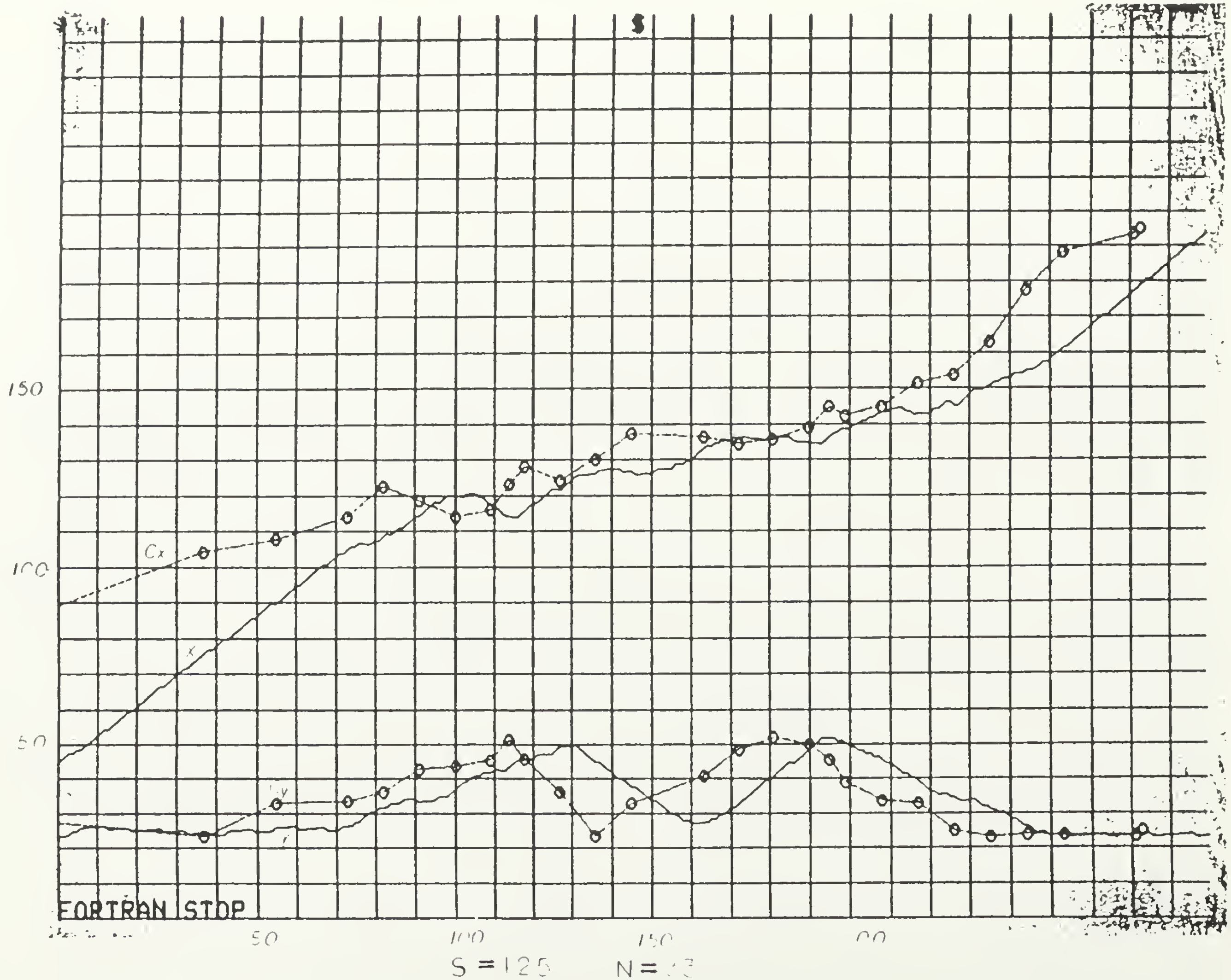


Figure 4.16 Plot X,Y,Cx, and Cy vs Z for a CGN at a Range of 45 K-ft.

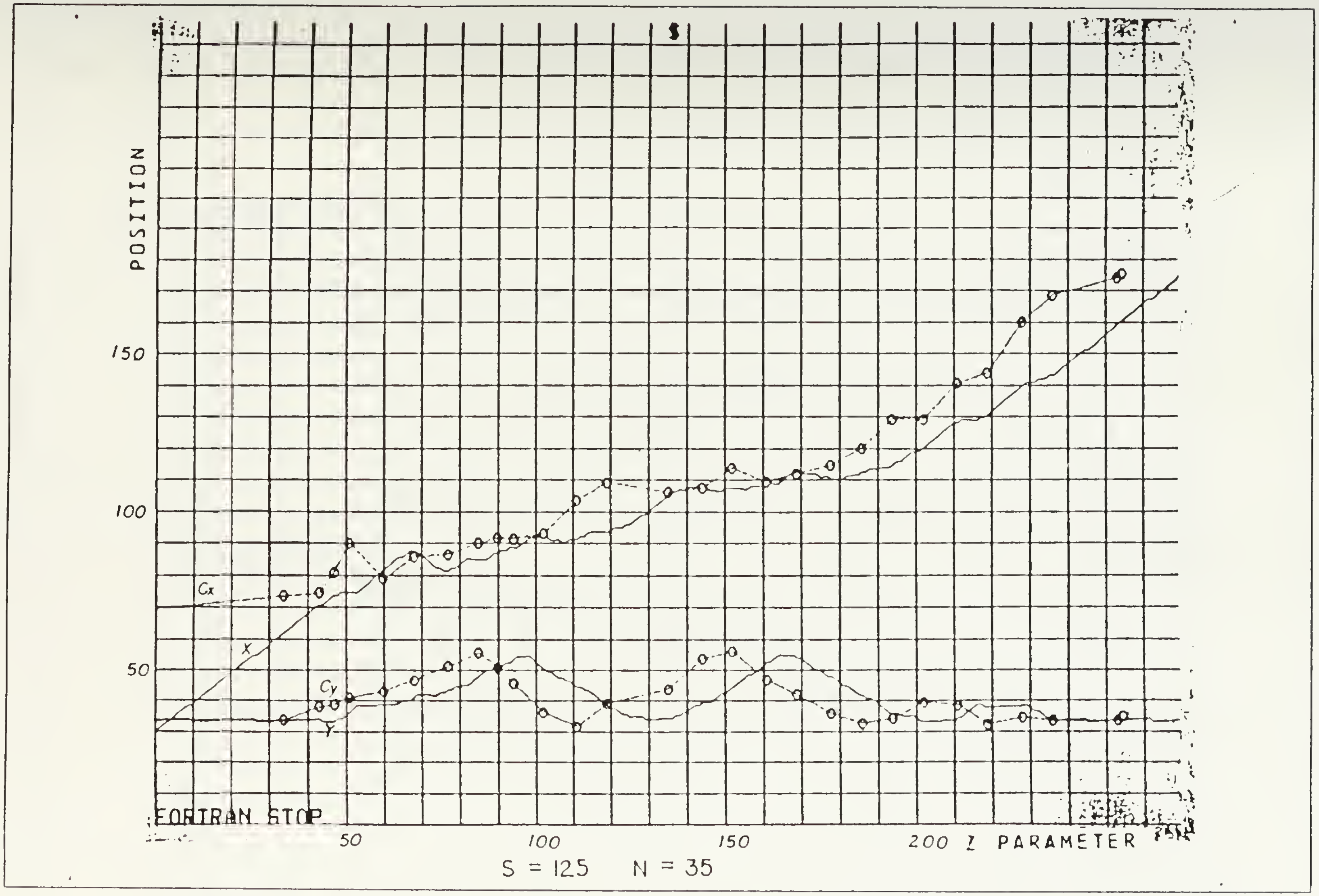


Figure 4.17 Plot X,Y,Cx, and Cy vs Z for a DDG at a Range of 41 K-ft.

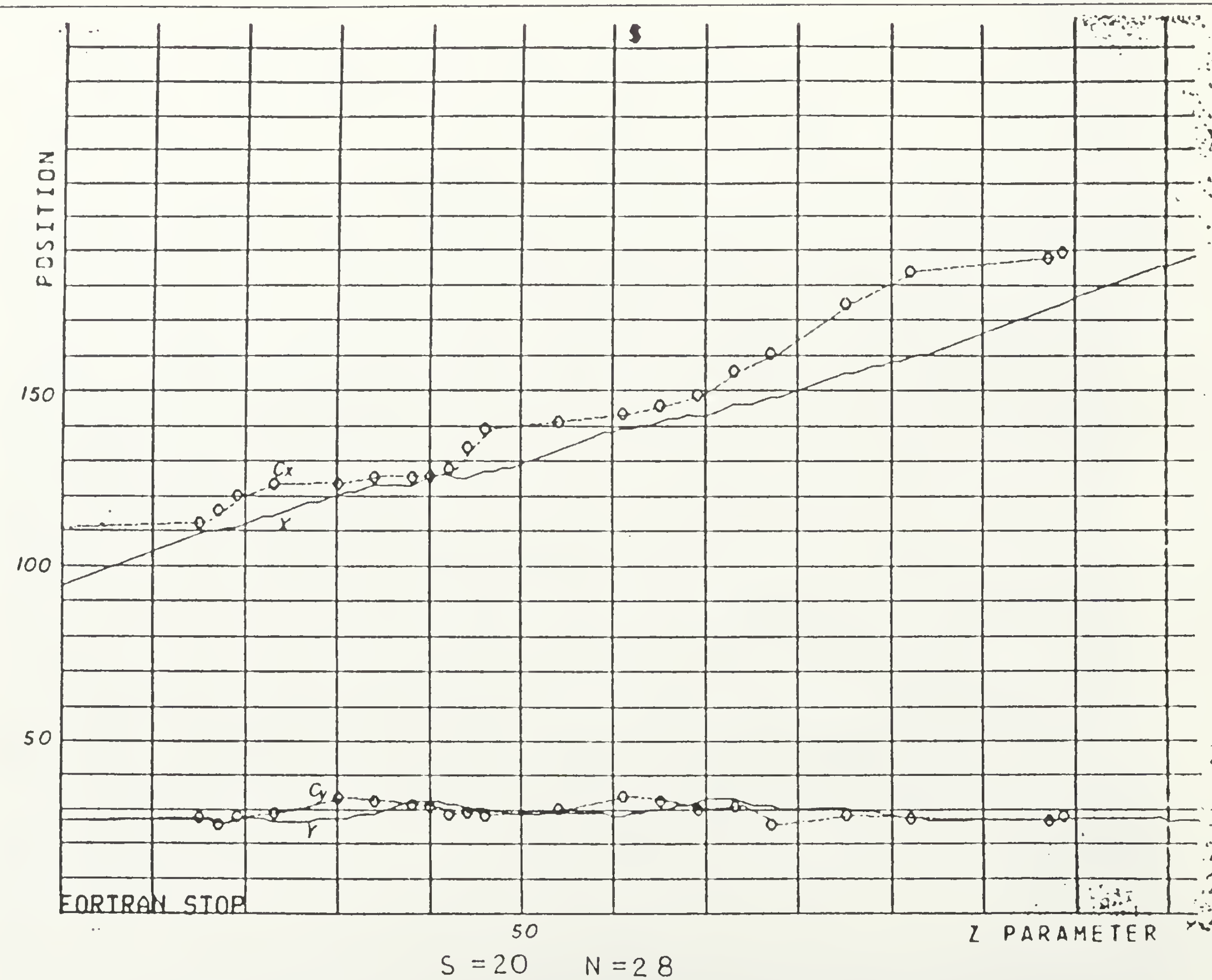


Figure 4.18 Plot X,Y,Cx, and Cy vs Z for a DD at a Range of 77 K-ft.

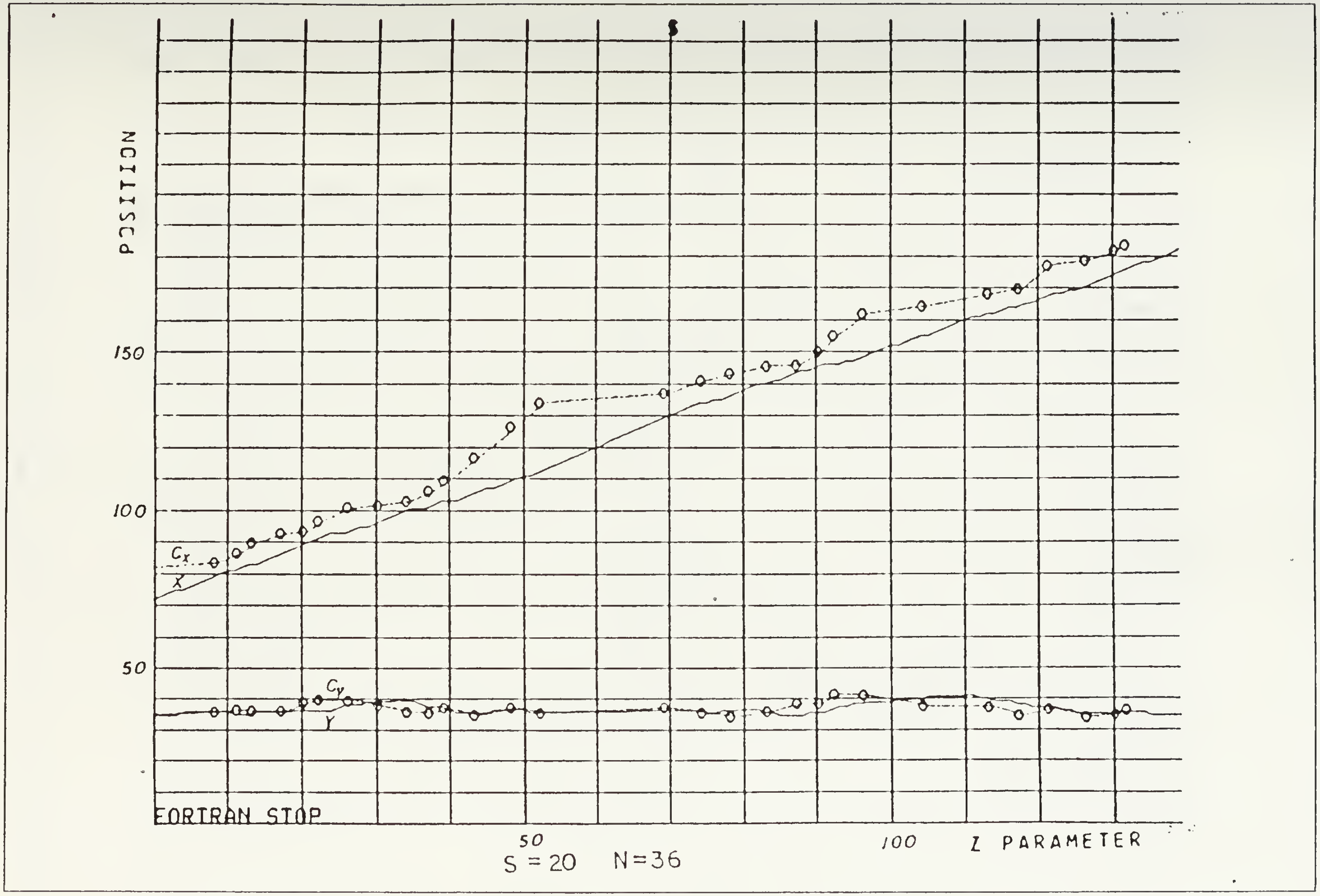


Figure 4.19 Plot X,Y,Cx, and Cy vs Z for a AOR at a Range of 78 K-ft.

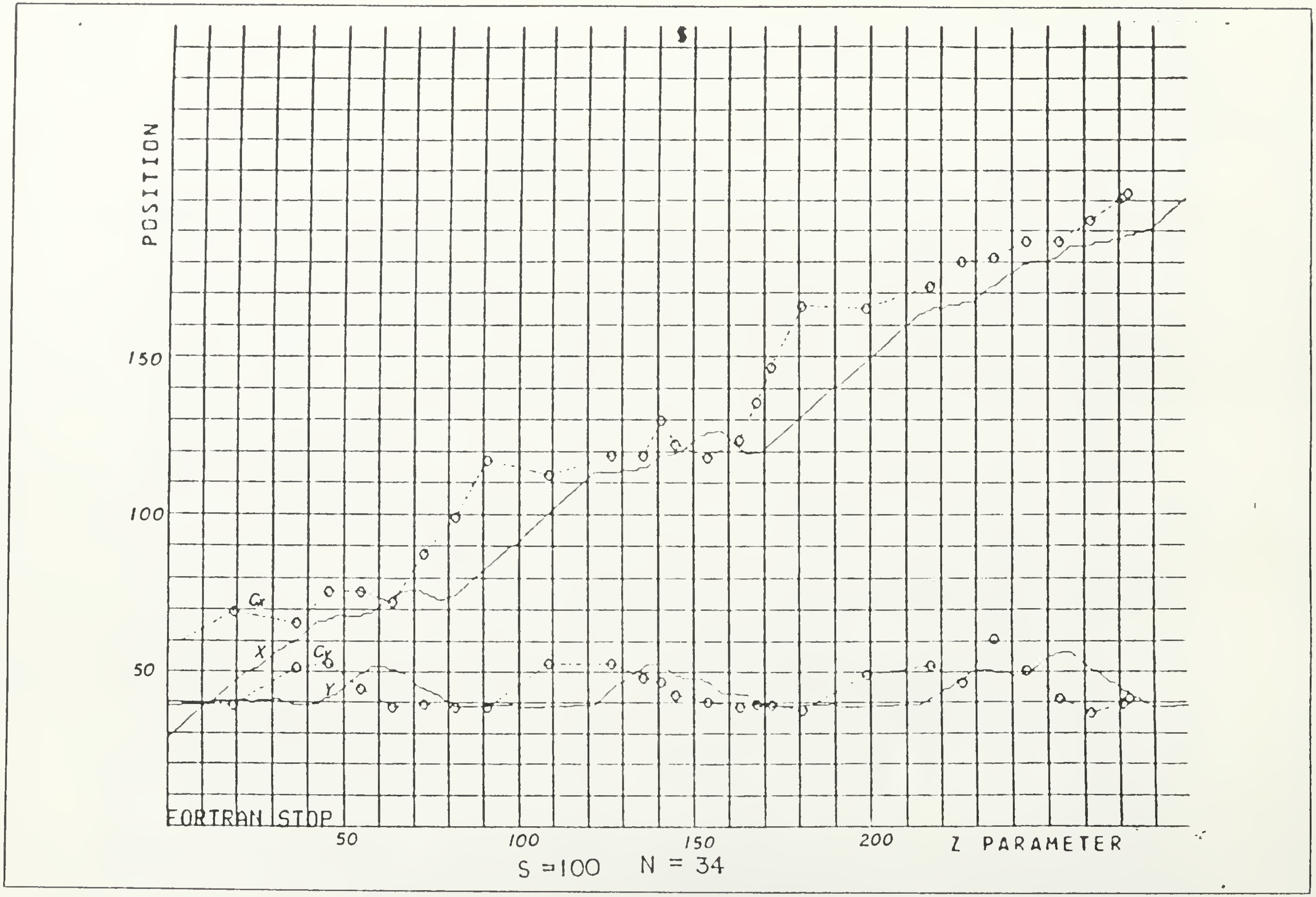


Figure 4.20 Plot X,Y,Cx, and Cy vs Z for a Freighter at a Range of 40 K-ft.

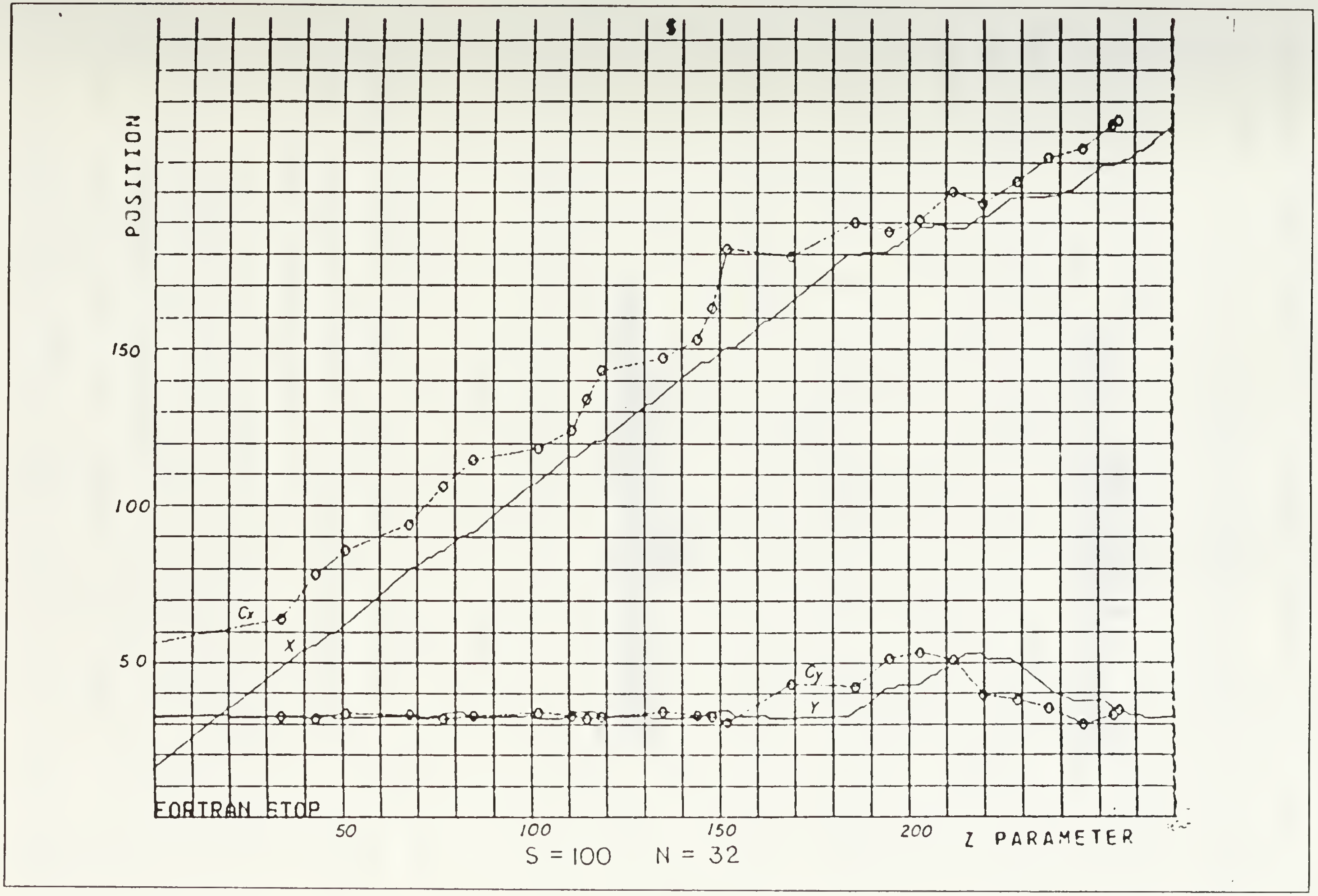


Figure 4.21 Plot X,Y,Cx, and CY vs Z for a Container at a Range of 28 K-ft.

D. SHIP DESCRIPTION

The shape of the ships depend upon the shapes and positions of the lumps. Characteristics of the lumps for different type of ships are as follows:

1. Frigate - The beginning of the lump is at 1/6 of the ship length to the left side of the midships and the peak of the lump is at the mast which is located at the midships. The termination of the lump is approximate 1/3 of the ship length from the midships to right side. In addition, the average height of the lump is a little higher than the level between the bow and the stern, and its size is 1/2 of the ship length as shown in Figure 4.22.

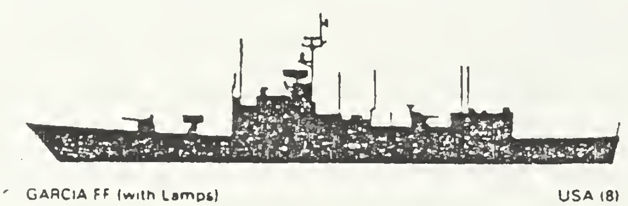


Figure 4.22 Frigate.

2. Container - The beginning of the lump is at 1/3 of the ship length to the right side of the midships while the lump is high and terminate at the stern. The lump appears to be in a rectangular shape with small crane.
3. Tank landing ship(LST) - The beginning of the lump is at 1/4 of the ship length from the midships to the left side; the height of the lump is higher than the level between the bow and the stern by a small margin, while its highest point is located at approximate 1/6 of the ship length from the midships

to the left side with small difference from the average high as shown in Figure 4.23.

Figure 4.23 Tank Landing Ship(LST).

4. Guided missile destroyer - the beginning of the lump starts at approximately 1/4 of the ship length from the midships to the left side with its highest point at the mast. After this the slope will decline in a very rapid fashion following a noticable deck distance, then the beginning of the second lump occurs due to the redome presence. Therefore, the lump will be narrow with great height, and will terminate at approximatly 1/6 of the ship length from the midships to the right side. Furthermore, there is a little lump near the stern, this distinguish destroyer of the same size as shown in Figure 4.24.
5. Destroyer-lump characteristic will be similar to that of guided missile destroyer except for the small lump as in Figure 4.25.
6. Guided missile cruiser - The beginning of the first lump is at 1/12 of the ship length from the midships to the left side with highest point at the mast. The lump size is large both in length and height and its height decreases to the point which is a little above the level between the bow and the stern. Therefore, the second lump begins with almost the same size as

Figure 4.24 Guided Missile Destroyer(DDG).

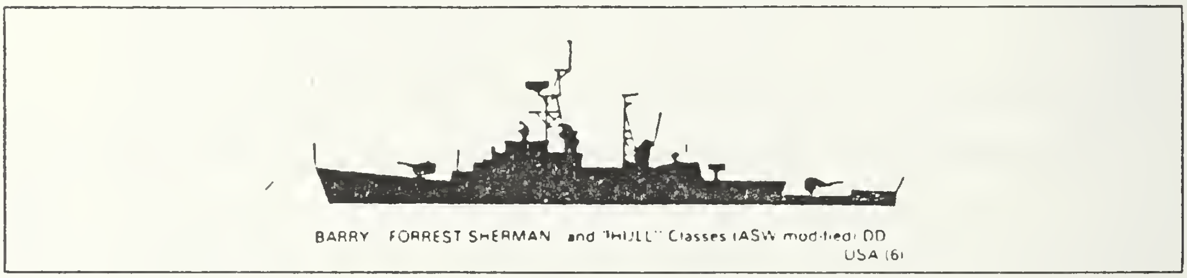


Figure 4.25 Destroyer.

the first one. The second lump ends at approximately 1/4 of the ship length from the midships to the right side. Furthermore, the distance between the peak of both lumps is less than that of the Guided missile destroyer shown in Figure 4.26.

Figure 4.26 Guided Missile Cruiser(CGN).

7. Replenishment oiler (AOR) - There are 2 little lumps with the first one starting at 4/5 of the ship length from the midships to the left side exhibiting rapid decrease in height to the level between the bow and the stern. The second lump starts at 1/4 of the ship length from the midships to the right side with slow decline to the point near the stern.

8. Freighter-with 3 noticeable lumps, the first two represent lifting crane with the first high lump approximately 1/6 of the ship length from the midships to the left side but narrow; meanwhile, the second lump is located at approximately the midships with the same size as the first one. The beginning of the third is at approximately 1/3 of the ship length from the midships to the right side with large size. It is higher than the first two and with gradual decline to the stern.

From the lump characteristic of different type of ships, we could distinguish the type of a ship based on the rotated superstructure.

E. SHIP CLASSIFICATION

An important aspect in categorizing types of ships is to recognize the lumps above the main deck. Therefore, it is useful to plot the positions of X and Y vs the Z parameter where the Z parameter can be determined from eq(4.5) where Z(I) is an array (1..M) as shown in Figure 4.27 where X is ploted in solid line and Y is ploted in dash line. Then, plot the B-spline coefficients Cx and Cy vs T(N); the knots position in Z where N is the total number of knots as shown in Figure 4.28. The Cx is ploted in solid line. The Cy is ploted in dash line. The Values of Cy will be close to the curve of Y while the values of Cx for the same knot posi-

tions as Cy is exhibiting monotonic increasing trend. When the difference in Cx is decreasing or zero, the value of Cy will have a pronounced change where the beginning or the ending points of a lump can be detected. The value of knot position(T) at those points related to the sizes of the Cx,Cy can be determined. Finally, with the values of Cx and Cy at those points known, the area of the lumps can also be determined.

Hence, from the ship's characteristics and information derived from the above procedure, classification of ships can be made by considering

First, the number of lumps detected, 1, 2, or 3.

1. The number of lump=1: Frigate, Tank landing ship
2. The number of lump=2: Destroyer, Guided missile cruiser, Guide missile Destroyer, and Replenishment oiler(AOR)
3. The number of lump=3: Freighter and Container

Second, the position of a lump relative the midships is measured. This quantity is scaled by the total length of the ship. This scaled quantity will be invariant with respect to the different ship sizes at different ranges

Third, the area of the lump is normalized to the ship length squared.

1. Lump Detection

As shown in the plot of X, Y, Cx, and Cy vs Z parameter in Figure 4.15, when the difference (ΔCx) between successive value of Cy varies from increasing to decreasing, and then, to increasing again, Cy exhibits noticable variation. From observation of Cx values, it is seen that the difference (ΔCx) always has variation in the same sequence as stated above. Therefore, only Cy values are taken into consideration in the program that detects lump.

There are two distinct procedures used in dealing with two different kinds of lumps big or small. Thus, it is necessary to distinguish in the first place whether the lump is big or small. For the big lump, the differences of Cy is positive for all initial four knots. It continues to the peak and decreases toward the ending of the lump. For the small lump, the difference of Cy is positive for the first two knots and stay constant or positive for the third knot, but the difference of Cy for the forth knot is negative, thereafter, the program proceeds to establish the following values for each lump detected:

First, the knot positions (Z value) at

1. the beginning of the lump
2. the ending of the lump

Second, the Knot number for

1. the beginning of the lump
2. the ending of the lump

Third,Number of lumps detected.

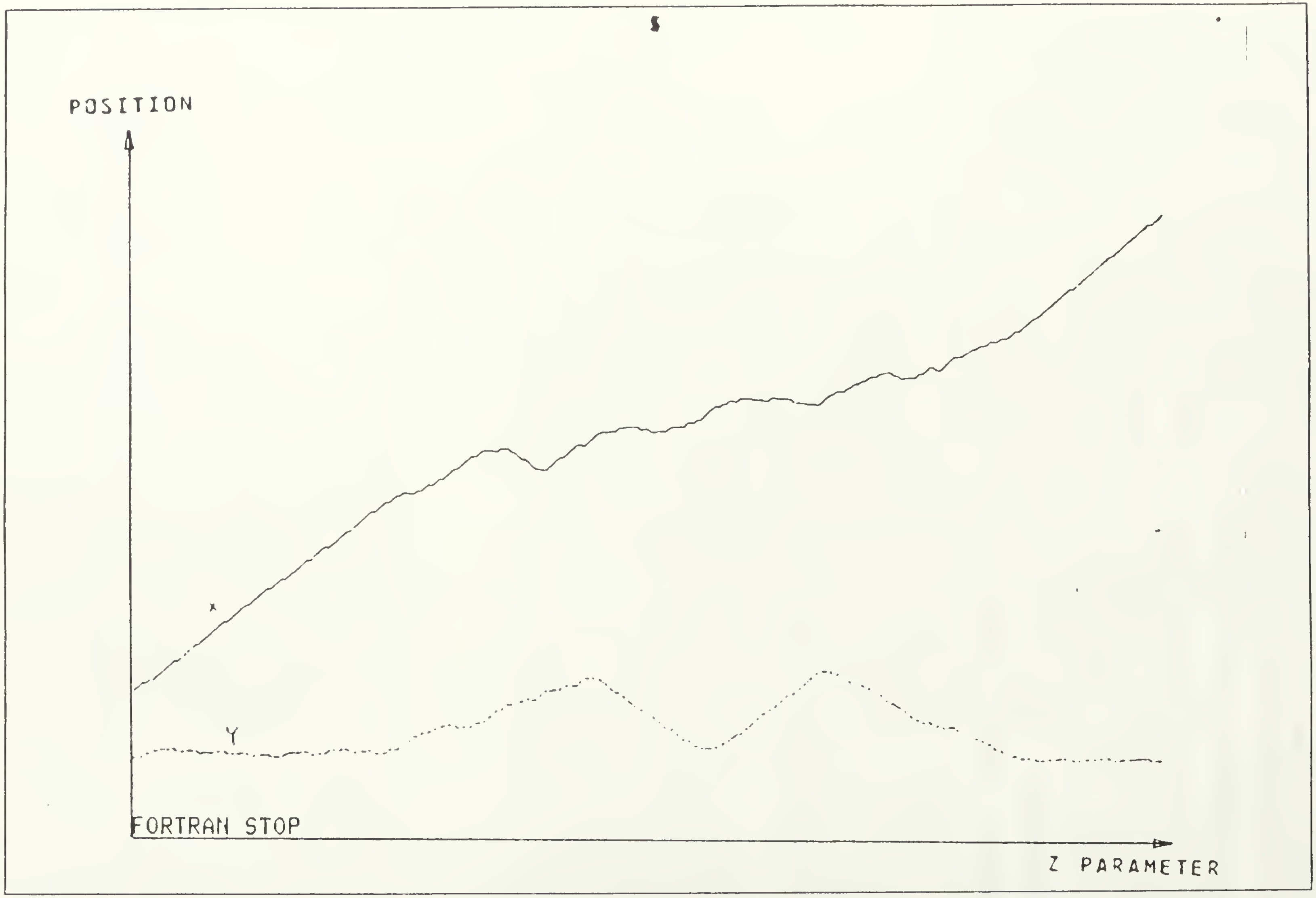


Figure 4.27 Plot X,Y vs Z for a CGN at 45kft.

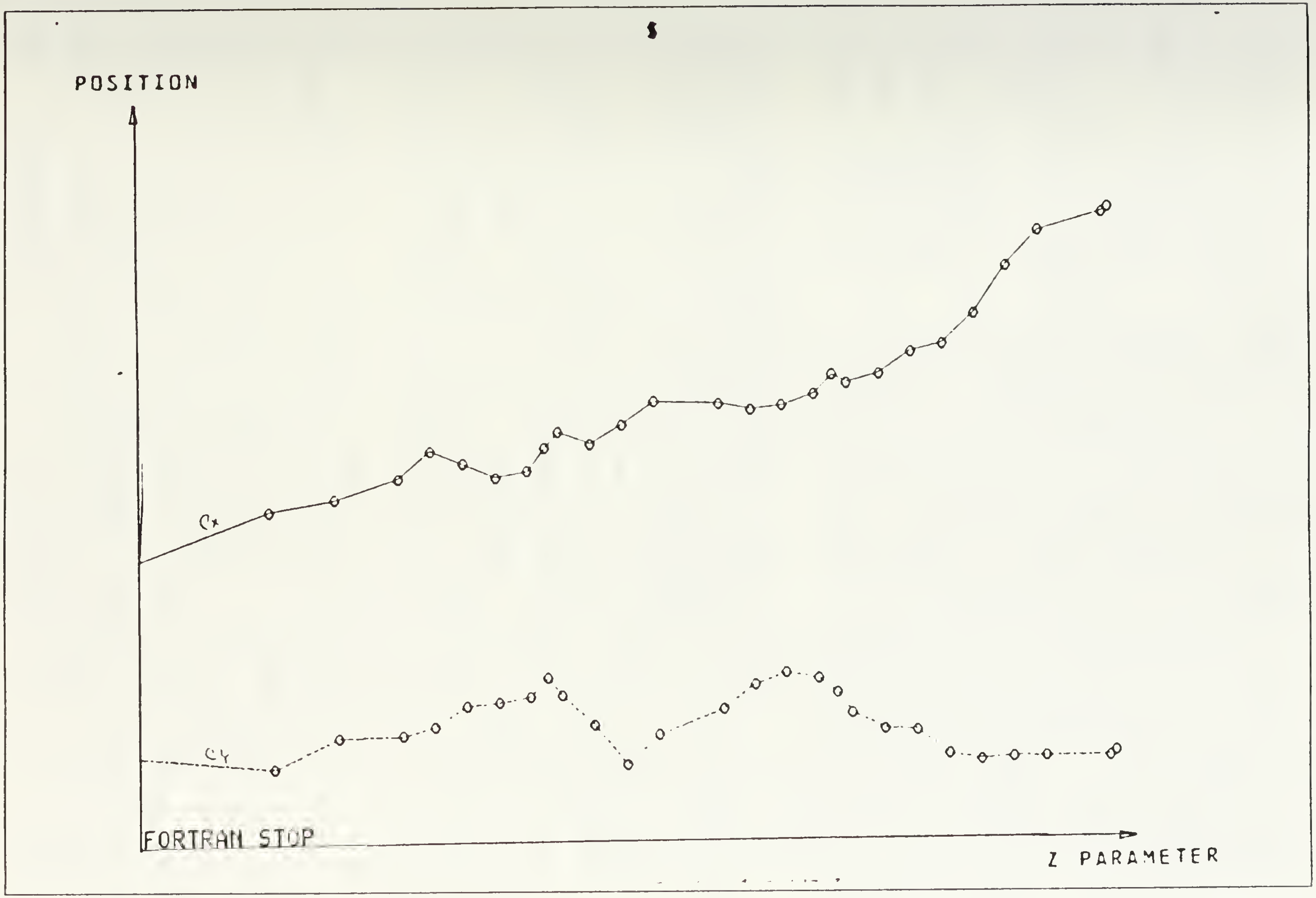


Figure 4.28 Plot Cx,Cy vs Z for a CGN at 45kft.

a. Procedure to Detect the Lump

1. Check a big or small lump by testing the conditions of three varying values of Cy increments (Δ Cy) in sequence. If they are positive it is a big lump and set the flag-lump to 1, otherwise it is a small lump and set the flag-lump to zero; then go to 2
2. Check the present knot position to see where its Z value equal to zero or to maximum Z value, if it is Z maximum then stop, it not go to 3
3. If the process begins at the first knot position, set the begin and the flag-end to zero; check the status of the flag-lump for 1 (big lump) or 0 (small lump), if it is a big lump, then go to 4. If it is a small lump, then go to 7.
4. Check the status for the begining or ending of the lump. If the flag-begin is 1, it represents that the beginning of a lump is found, then go to 5. Otherwise it's not found, then go to 6.
5. Find the ending of the big lump by testing the conditions of 3 values of Cy increments in sequence. The first 2 should be negative and the third should be constant or positive. If the condition are satisfied, store the Z value of the position of the third knot, and the flag-begin to zero; then go to 10.
6. Find the beginning of the big lump by testing the conditions of 3 different values of Cy increment (Δ Cy) in sequence. The first Cy should be negative or constant, the second should be positive, and the third should be positive. If the conditions are satisfied, store the Z value of the position of the second knot, and set flag-begin to 1; then go to 10.
7. Check the status for the beginning or ending of the lump. If the flag-begin is 1, it represents the

beginning of a lump is found, then go to 8. Otherwise it's not found, then go to 9.

8. Find the ending of the small lump by testing the conditions of 3 values of Cy increment (Δ Cy) in sequence. The first one should be negative or constant. If the conditions are satisfied, store the Z value of the position of the second knot, and set the flag-begin to zero; then go to 10.
9. Find the beginning of the small lump by testing the conditions of 3 values of Cy increment (Δ Cy) in sequence. The first should be constant or negative, the second should be positive, the third should be constant. If the conditions are satisfied, store the position of the second knot, and set the flag-begin to 1; then go to 10.
10. Move to the next knot, then go to 2.

This procedure is shown in Figure 4.29 and the detail of each procedure is shown in Appendix D.

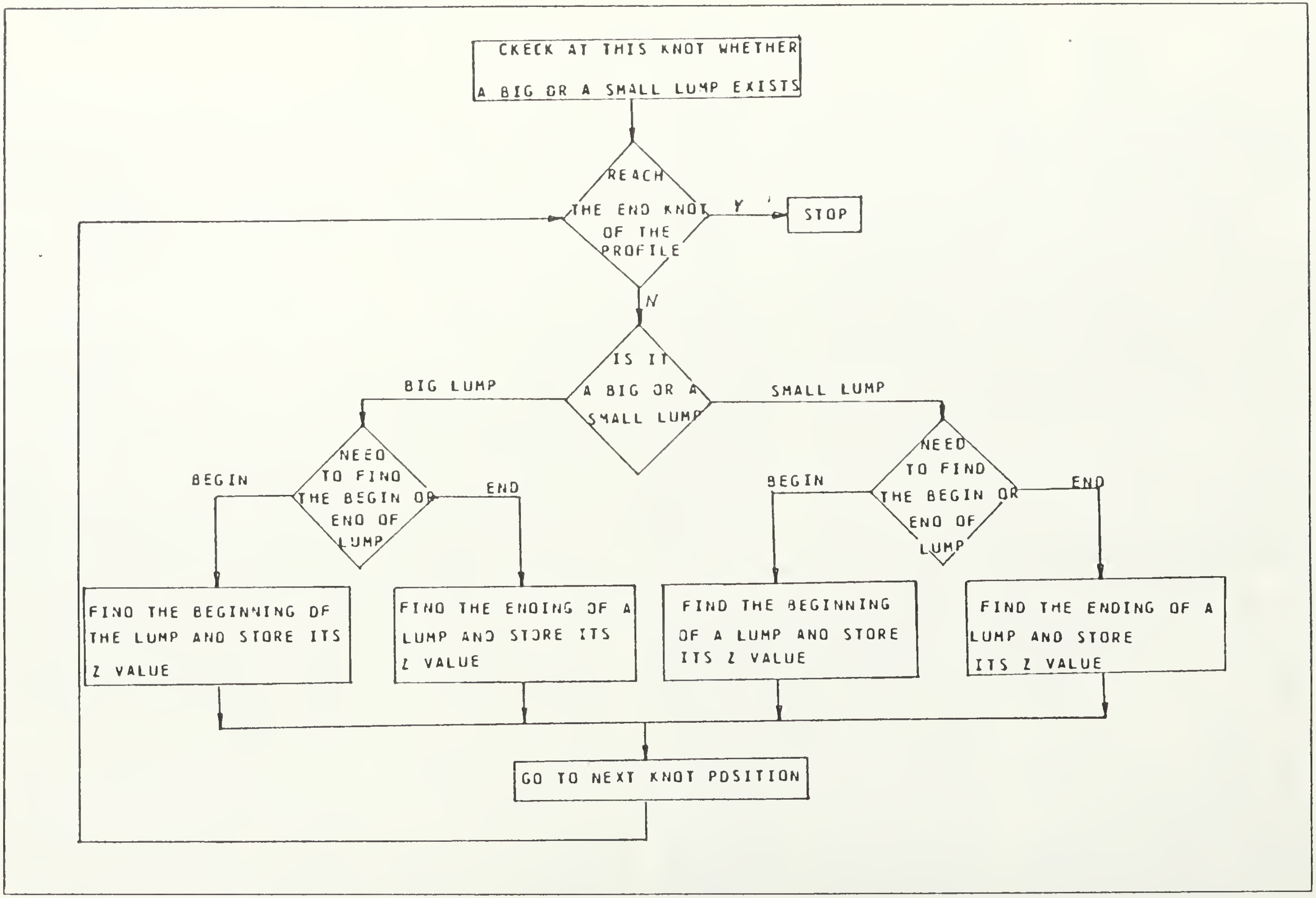


Figure 4.29 Flow Chart for Detecting Lumps.

2. To Determine the Area Under a Lump

The area under the lump can be determined by

$$\Delta AREA = \frac{\Delta C_x \Delta C_y}{2} + C_y \Delta C_y \qquad (4.10)$$

Suppose there is a lump as shown in Figure 4.30. The area increment can be calculated by using eq(4.10). Then the area under BC(which is negative) is added to the area under AB.

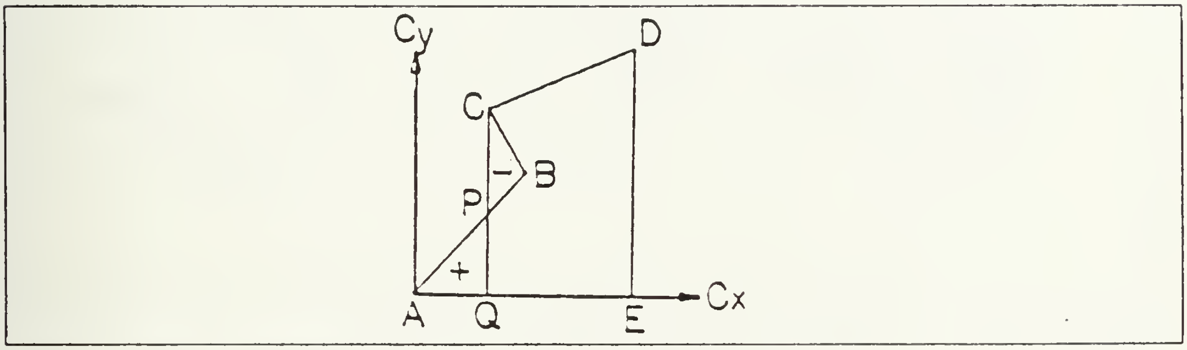


Figure 4.30 First Procedure to Determine the Area.

Next, area under CD is calculated, which is positive and add this to the last resulting area. Obviously, the sum would represent the total area of the lump as shown in Figure 4.31.

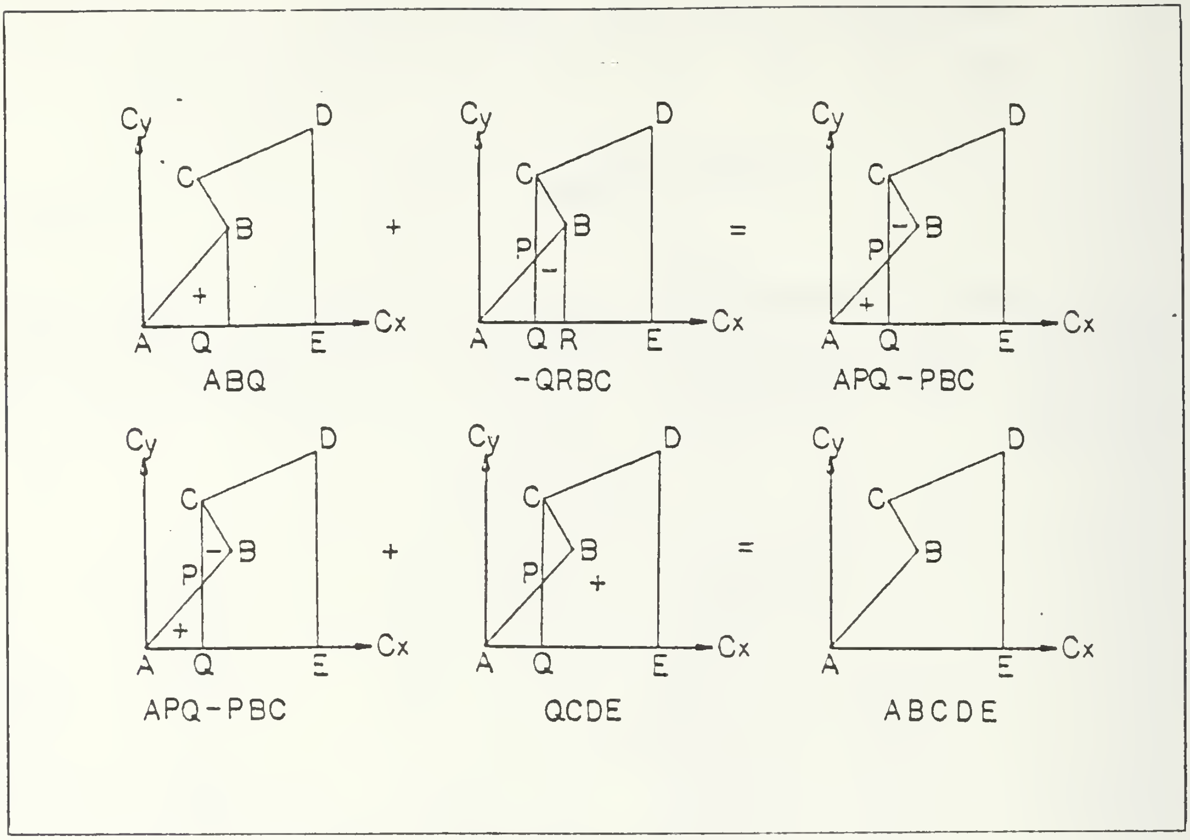


Figure 4.31 Step by Step Procedure to Determine the Area.

F. CONSTRUCTION OF THE DECISION TREE

The characteristics of ships used for classification are, the number of lumps, the area of lumps, the knot positions(Z value) and where the beginning of the lump and lump maxima relative to the midships are located.

In view of the above characteristic, it is necessary to find the relationships among them to make classification possible. Therefore, for each of the eight different class of ships, the decision tree is constructed which is based upon relationships observed in the plots of the knot position (Z values) for the beginning of the lumps normalized by the total ship length (Z value) VS. the number of lumps; the area of the lumps normalized by the total ship length

(Z value) squared vs the number of lumps; and the Z value of the lump maxima, as shown in Figure 4.32 through Figure 4.39.

Thus, according to the number of lumps presented in the profile used as the first criteria, ships can be divided into 3 distinct groups. Then, classification, can be made by comparing further characteristic as shown in Table III, and the complete decision tree constructed from Table III is shown in Figure 4.40.

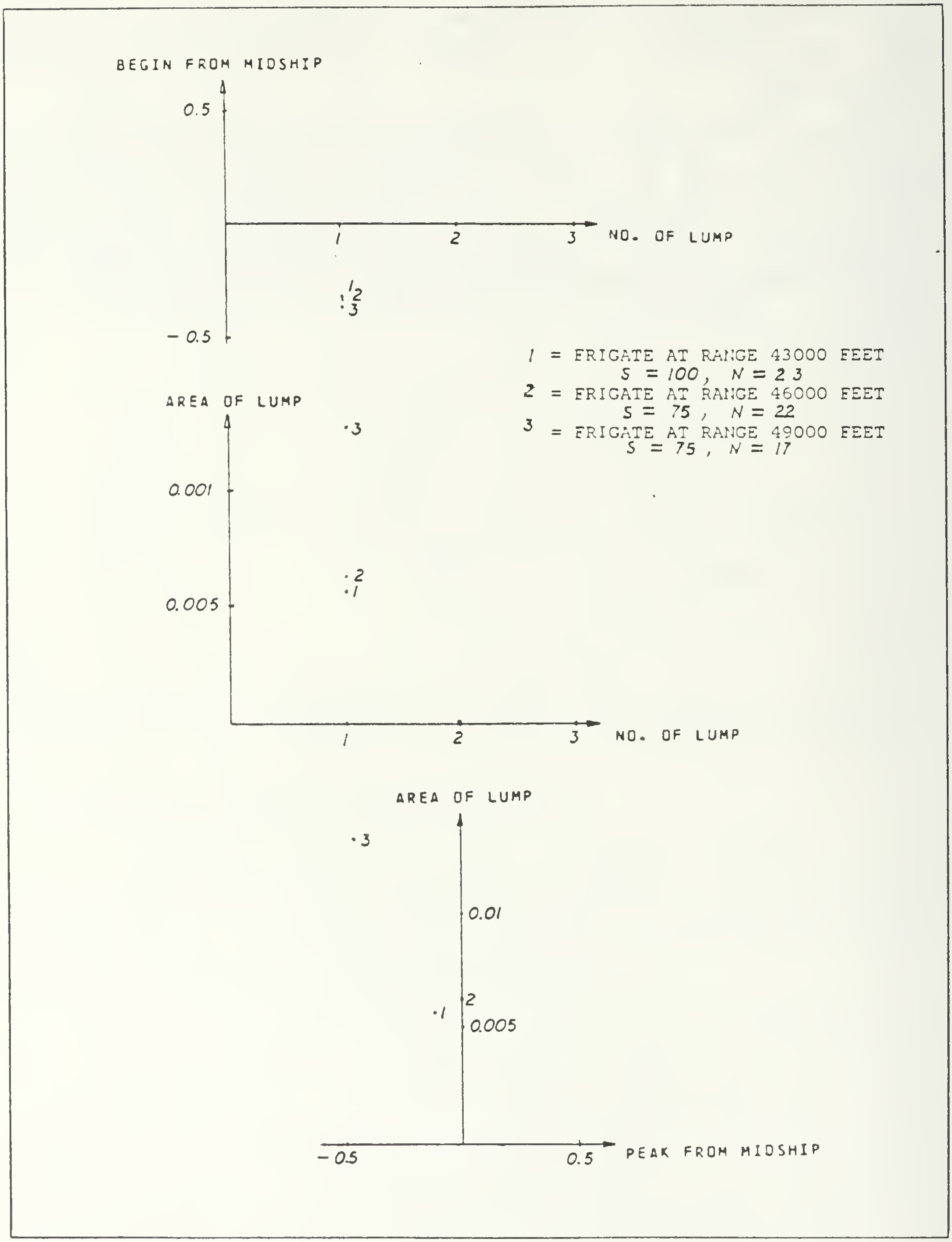


Figure 4.32 Plot the Beginning, Area, and Peak of a FF.

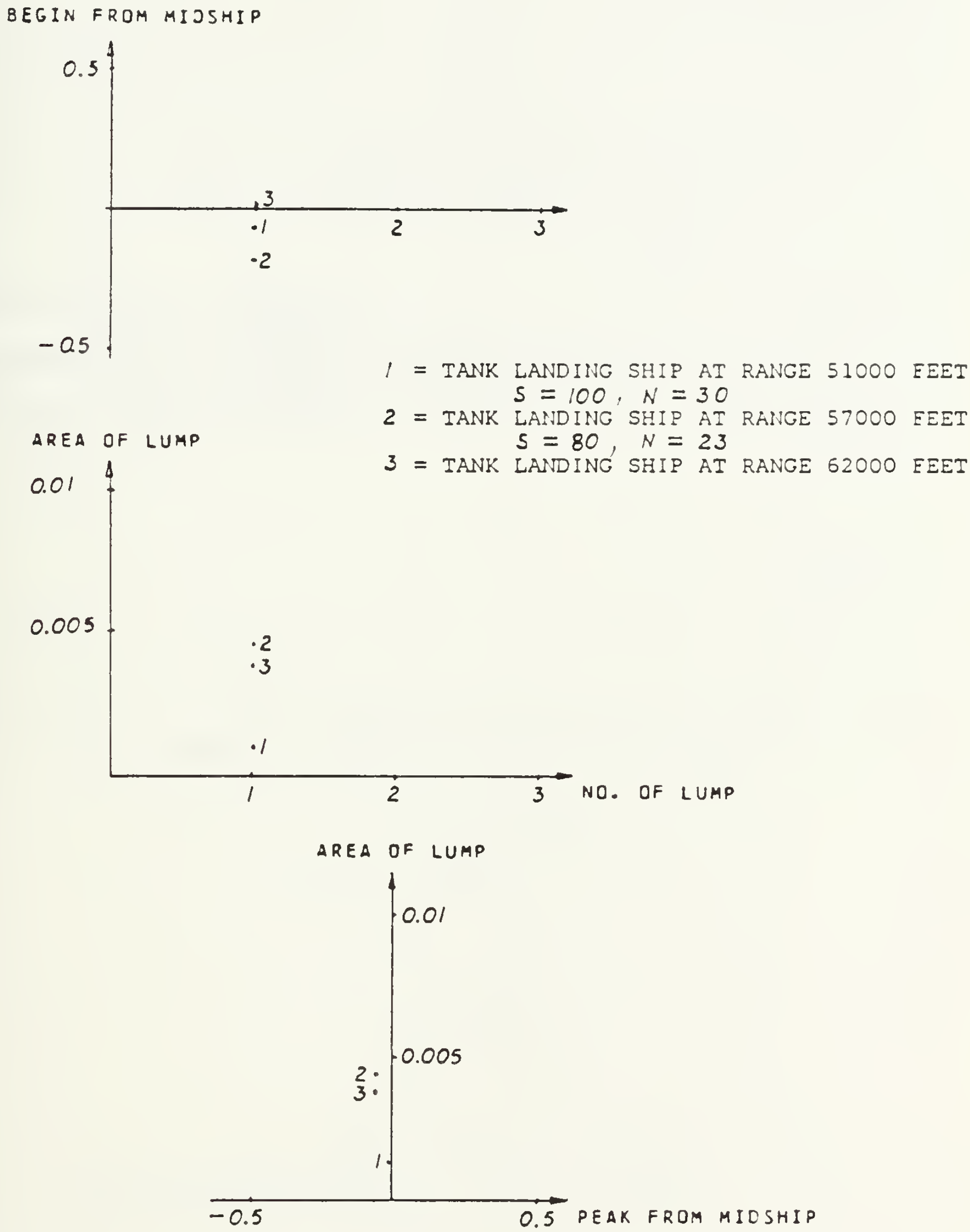


Figure 4.33 Plot the Beginning, Area, and Peak of a LST.

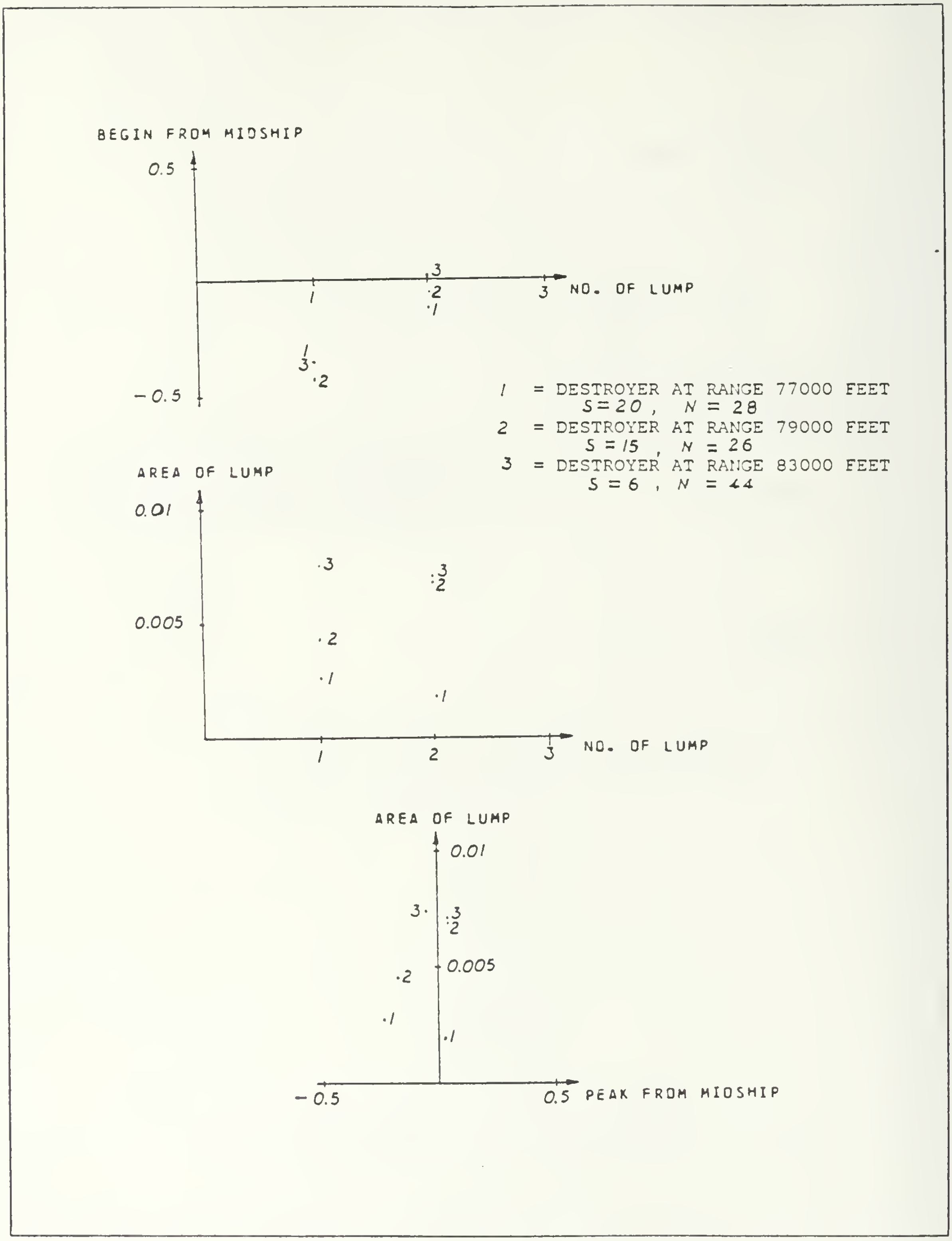


Figure 4.34 Plot the Beginning, Area, and Peak of a DD.

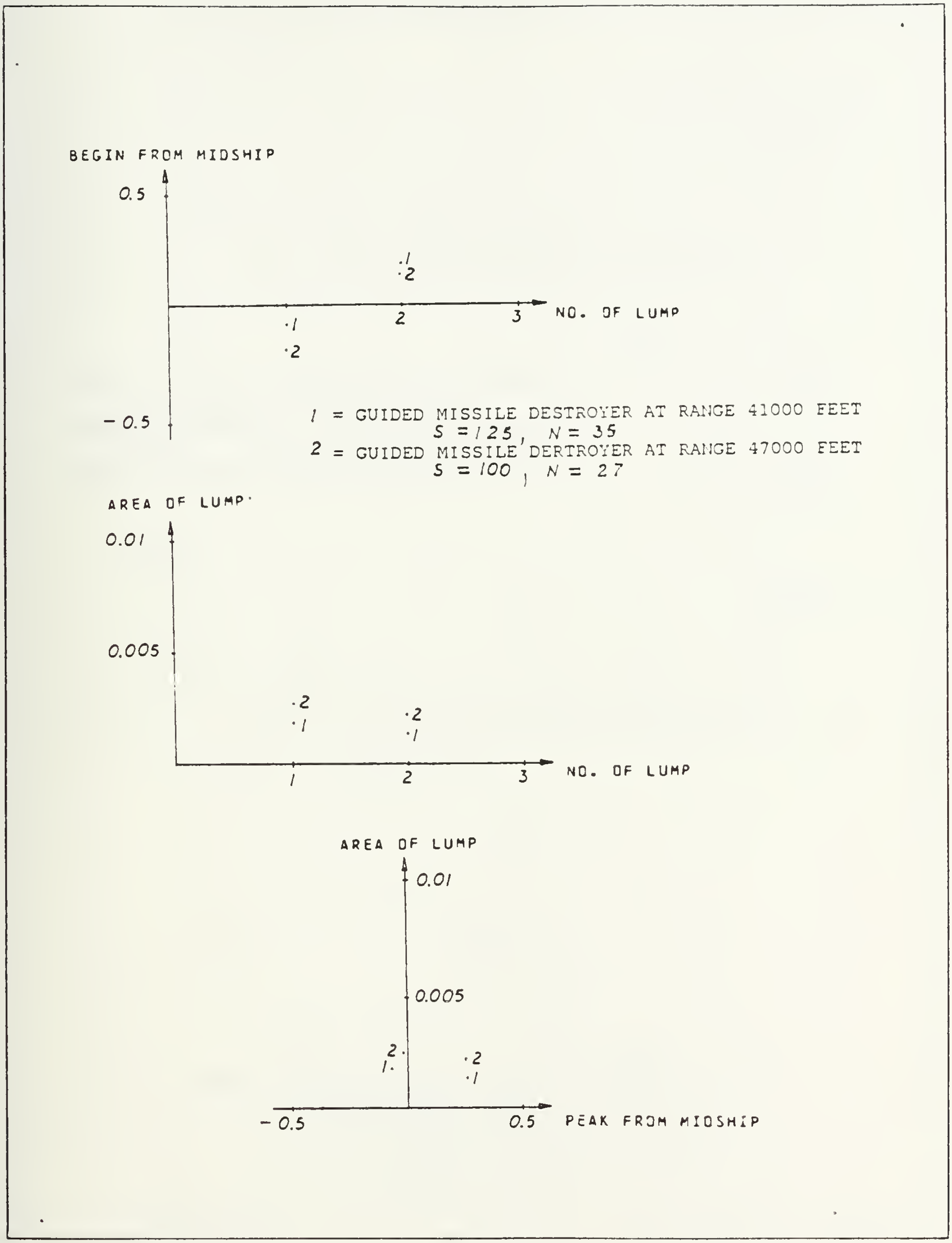


Figure 4.35 Plot the Beginning, Area, and Peak of a DDG.

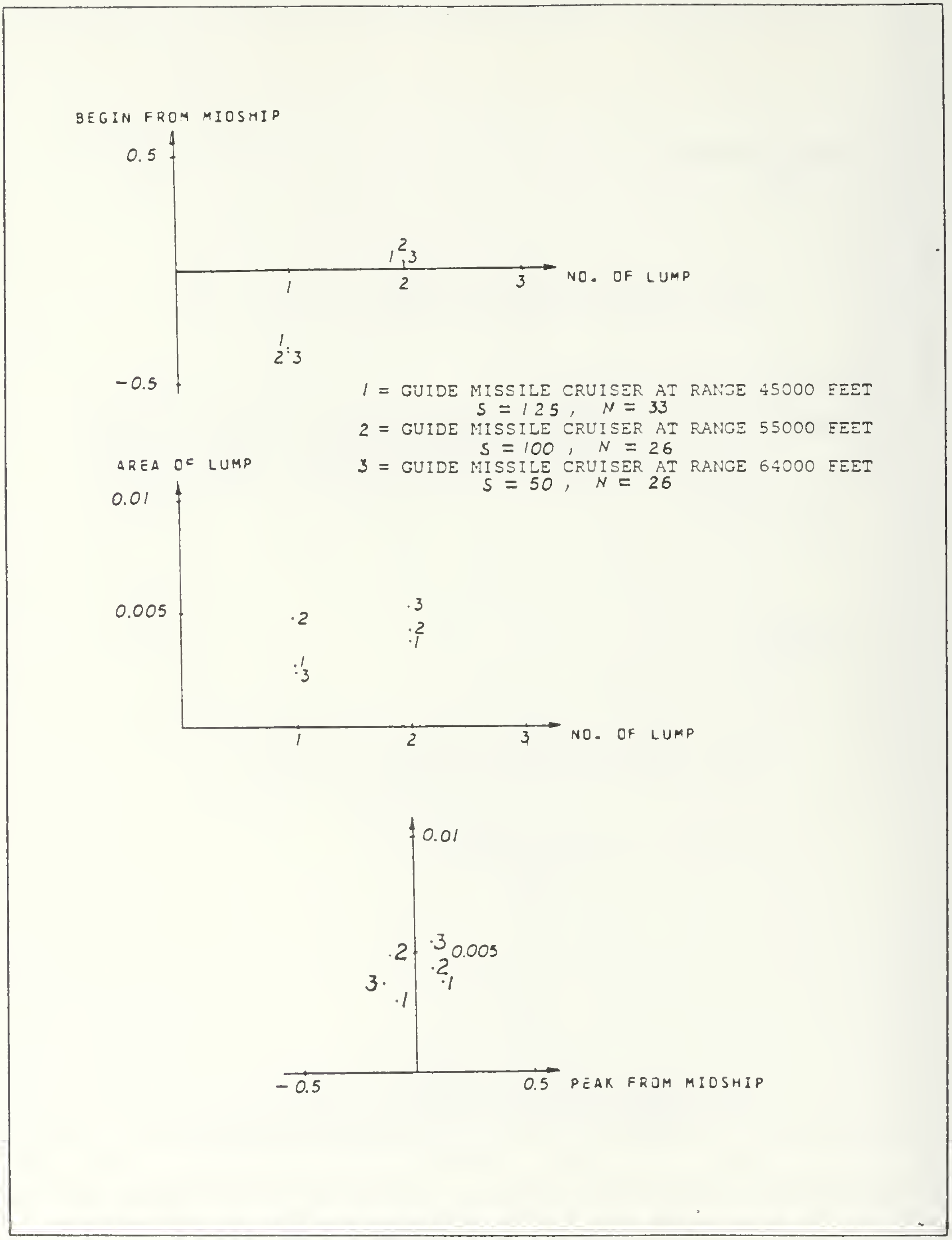


Figure 4.36 Plot the Beginning, Area, and Peak of a CGN.

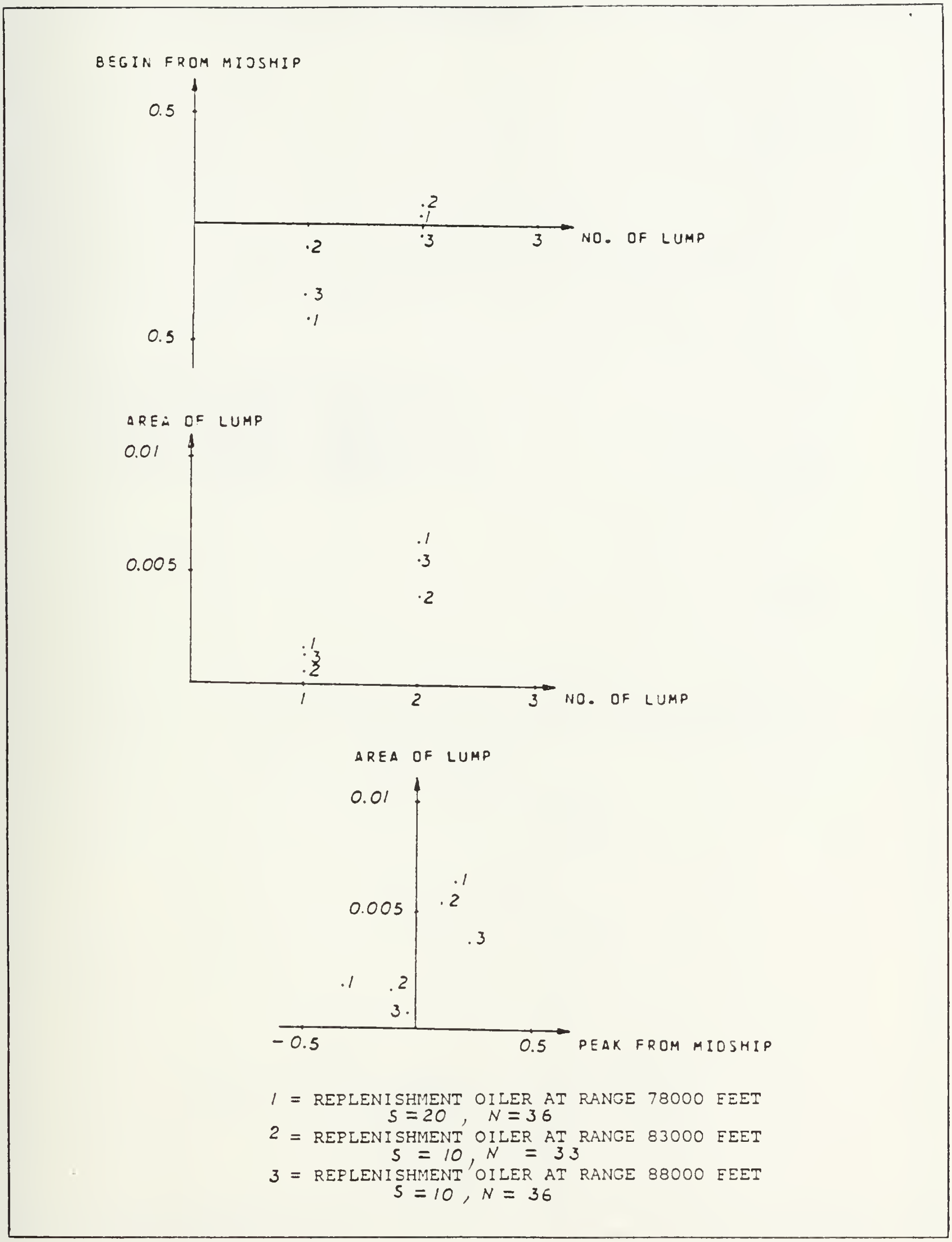


Figure 4.37 Plot the Beginning, Area, and Peak of a AOR.

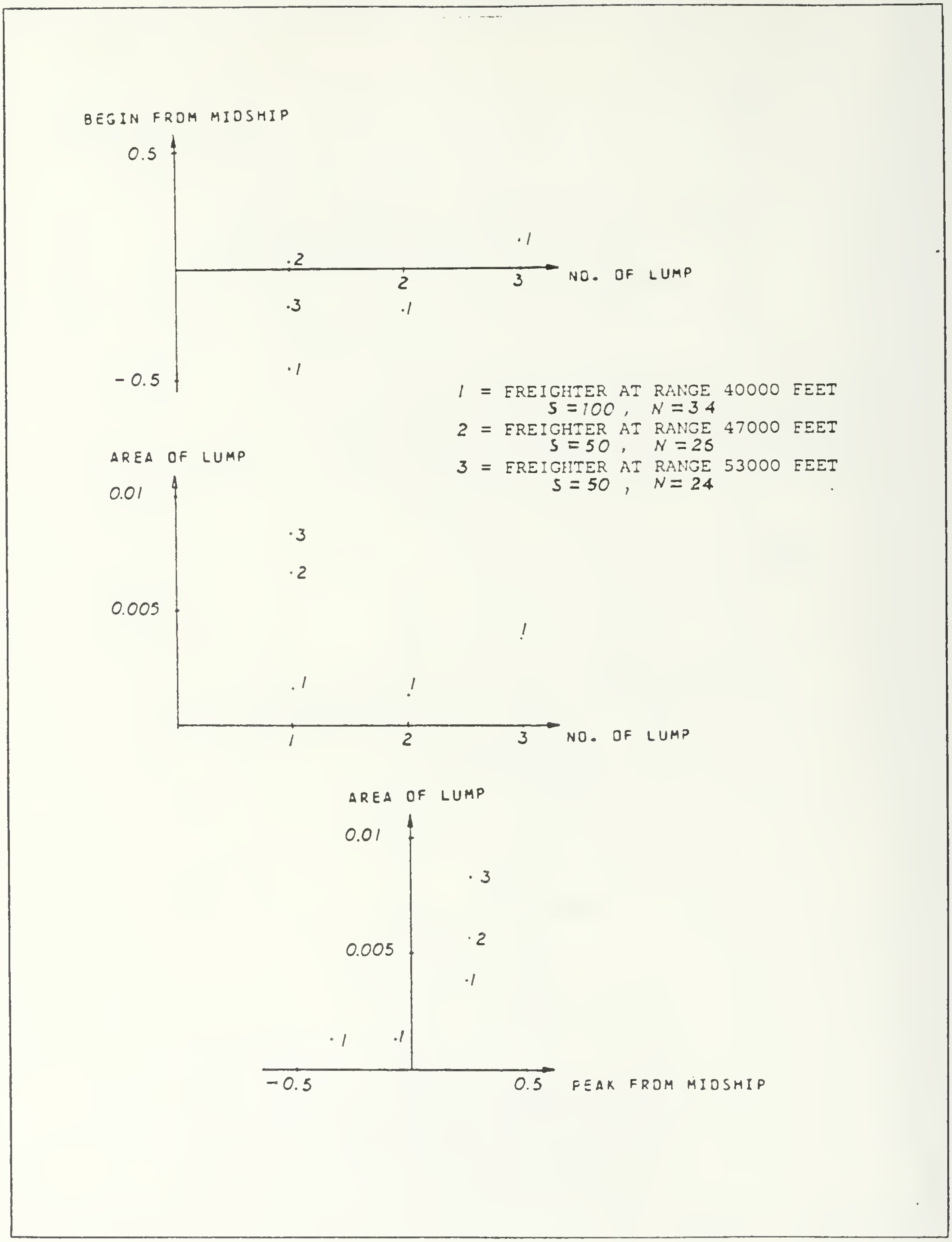


Figure 4.38 Plot the Beginning, Area, and Peak of a Freighter.

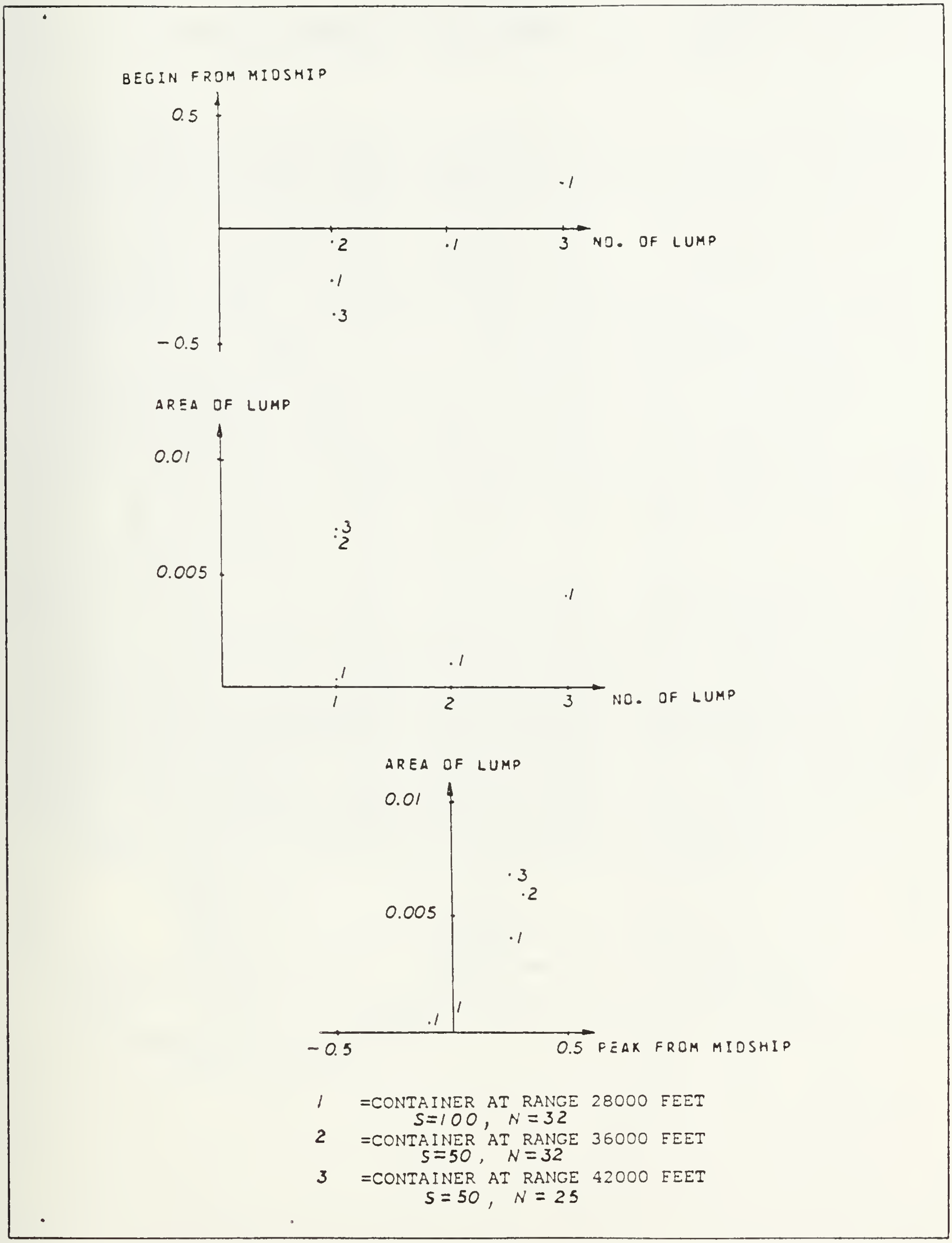
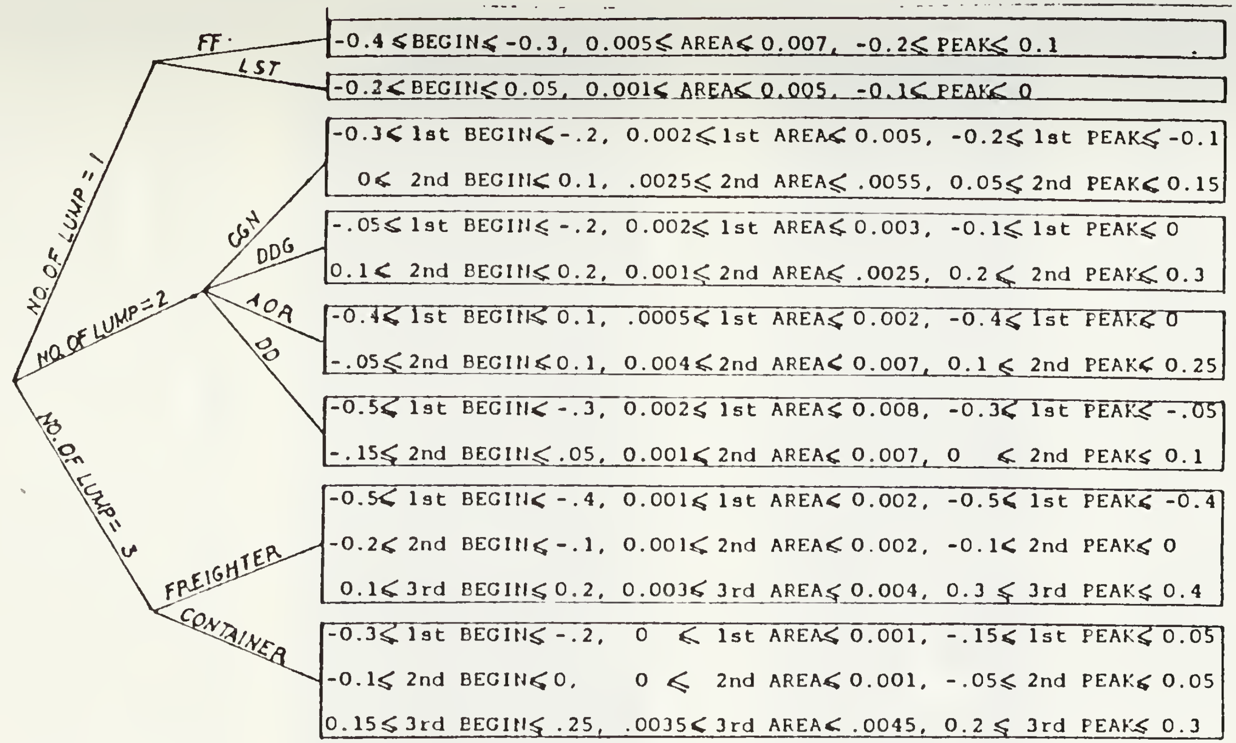


Figure 4.39 Plot the Beginning, Area, and Peak of a Container.

TABLE III

Comparison of Different Types of Ships

No. OF LUMP	BEGINNING OF LUMP FROM MIDSHIP	PEAK IN THE LUMP	AREA UNDER THE LUMP	RESULT
1	$-0.4 \le BEGIN \le -0.3$	$-0.2 \le PEAK \le 0.1$	$0.005 \le AREA \le 0.007$	FF
	$-0.2 \le BEGIN \le 0.05$	$-0.1 \le PEAK \le 0$	$0.001 \le AREA \le 0.005$	LST
2	$-0.3 \le 1st\ BEGIN \le -0.2$ $0 \le 2nd\ BEGIN \le 0.1$	$-0.2 \le 1st\ PEAK \le -0.1$ $0.05 \le 2nd\ PEAK \le 0.15$	$0.002 \le 1st\ AREA \le 0.005$ $.0025 \le 2nd\ AREA \le .0055$	CGN
	$-0.2 \le 1st\ BEGIN \le -0.05$ $0.1 \le 2nd\ BEGIN \le 0.2$	$-0.1 \le 1st\ PEAK \le 0$ $0.2 \le 2nd\ PEAK \le 0.3$	$0.002 \le 1st\ AREA \le 0.003$ $0.001 \le 2nd\ AREA \le .0025$	DDG
	$-0.4 \le 1st\ BEGIN \le 0.1$ $-0.5 \le 2nd\ BEGIN \le 0.1$	$-0.4 \le 1st\ PEAK \le 0$ $0.1 \le 2nd\ PEAK \le 0.25$	$.0005 \le 1st\ AREA \le 0.002$ $0.004 \le 2nd\ AREA \le 0.007$	AOR
	$-0.5 \le 1st\ BEGIN \le -0.3$ $-.15 \le 2nd\ BEGIN \le 0.05$	$-0.3 \le 1st\ PEAK \le 0.05$ $0 \le 2nd\ PEAK \le 0.1$	$0.002 \le 1st\ AREA \le 0.008$ $0.001 \le 2nd\ AREA \le 0.007$	DD
3	$-0.5 \le 1st\ BEGIN \le -0.4$ $-0.2 \le 2nd\ BEGIN \le -0.1$ $0.1 \le 3rd\ BEGIN \le 0.2$	$-0.5 \le 1st\ PEAK \le -0.4$ $-0.1 \le 2nd\ PEAK \le 0$ $0.3 \le 3rd\ PEAK \le 0.4$	$0.001 \le 1st\ AREA \le 0.002$ $0.001 \le 2nd\ AREA \le 0.002$ $0.003 \le 3rd\ AREA \le 0.004$	FREIGHTER
	$-0.3 \le 1st\ BEGIN \le -0.2$ $-0.1 \le 2nd\ BEGIN \le 0$ $0.15 \le 3rd\ BEGIN \le 0.25$	$-.15 \le 1st\ PEAK \le -.05$ $-.05 \le 2nd\ PEAK \le 0.05$ $0.2 \le 3rd\ PEAK \le 0.3$	$0 \le 1st\ AREA \le 0.001$ $0 \le 2nd\ AREA \le 0.001$ $.0035 \le 3rd\ AREA \le .0045$	CONTAINER



BEGIN = THE BEGINNING OF THE LUMP FROM MIDSHIP NORMALIZE BY THE TOTAL LENGTH OF THE SHIP

AREA = THE AREA OF THE LUMP NORMALIZE BY THE TOTAL LENGTH OF SHIP SQUARED

PEAK = THE PEAK HEIGHT OF THE LUMP NORMALIZE BY TOTAL LENGTH OF SHIP

Figure 4.40 The Decision Tree.

G. SUMMARY

The decision tree which is constructed from Table III does not provide 100-percent correct classification for all types of ship images. Furthermore, the presence of noise in images may cause complicatations in classifying the ships. For example, with excessive noise in the original image, Sobel operator for edge enhancement in the preprocessing process, still yield result with residual noise present in Figure 4.41. These residual noise is undesirable since it causes failure to extract profiles which retain necessary informations from the original images. The subsequent classification of profile by the Fourier Coefficient method and the B-spline Coefficient method becomes difficult.

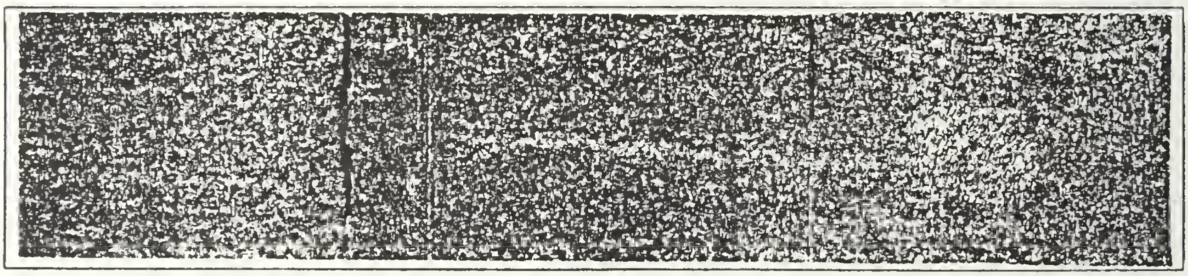


Figure 4.41 Noisy Image.

As discussed before in order to reduce the noise, an appropriate threshold gray value for the preprocessed image is set. This results in a silhouette image. However, if the value is too high the profile becomes broken which prevent successful operation in the closing process discussed in chapter 2, as shown in Figure 4.42. On the other hand decreasing the threshold value results in erroneous profile, as shown in Figure 4.43. In some cases, when the closing process is used, certain vital information is lost. This effect can be seen in comparing the image in Figure 4.44,

and Figure 4.45. Therefore, loss of informations due to the attempt of eliminating noise and the inability in the preprocessing process to cope with the residual noise, yield erroneous profile. Consequently, failure occurs in the succeeding classification steps.

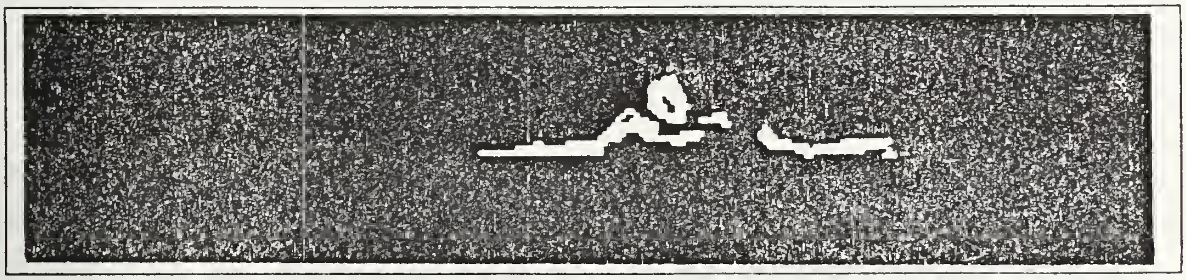


Figure 4.42 High Threshold.

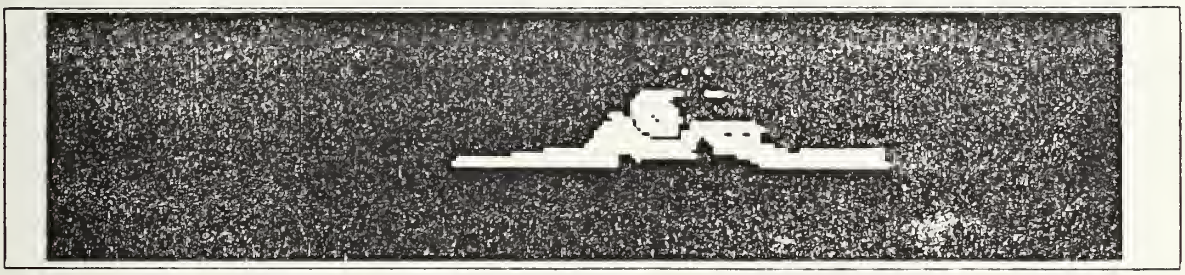


Figure 4.43 Low Threshold.

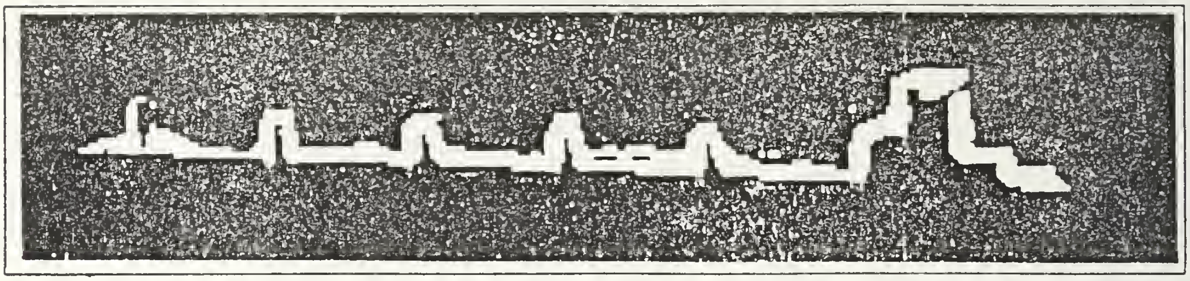


Figure 4.44 Before Closing.

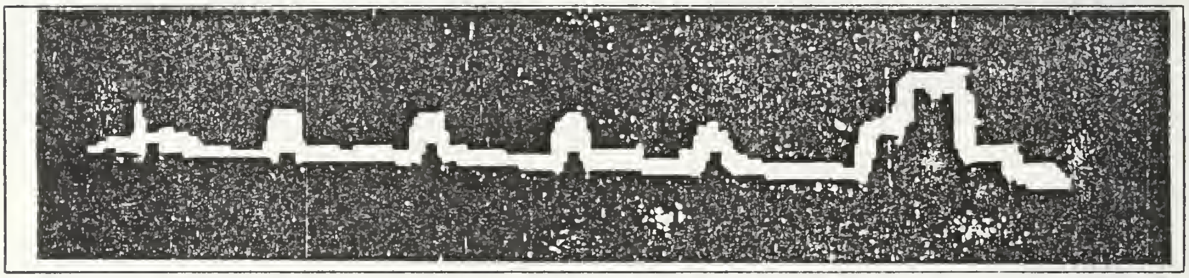


Figure 4.45 After Closing.

V. CONCLUSION

The shape of the superstructure of a ship is the most important feature used in the classification algorithms. To extract the edge profile, a Sobel operator is employed. The limitation of a Sobel operator is that some small details of the edge is lost. For example, the image of a DD at a range of 79000 feet after applying the Sobel operator shows the superstructure and the radar. But the edge of a small mast disappeared as shown in Figure 5.1. Furthermore, if the threshold value is set too big, the edge image of the superstructure profile becomes broken. The top superstructure profile is obtained by setting the gray value under the slope between the bow and the stern to zero. We need to apply a contour tracking process to refine the superstructure profile. For some images, the connection of the broken profile pieces may be achieved in a Closing operation as discussed in Chapter 2. The disadvantage of the Closing operation is that some small details of the profile may disappear. The superstructure profile of a freighter at a range of 53000 feet is shown in Figure 5.2. After the Closing process the detail of the cranes disappeared as shown in Figure 5.3.

The ship classification can be achieved by either the Fourier Coefficient method or the B-spline Coefficient method. In using these coefficients to classify ships, it is found that only the initial coefficients that lie between the 0th to the 20th, are relevant, while the rest are not. Inspection of the comparison curves of the same class of ships shows that, similarities in patterns exists up to the 20-th point. Beyond that, diversities in shape are so great that inclusion of those additional points for classification

will be of no use. Examples are shown in Figure 5.4 and Figure 5.5.

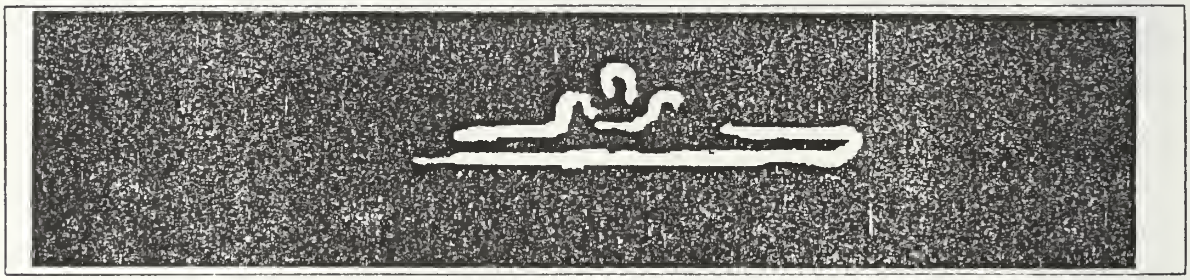


Figure 5.1 Edge Image of a DD at a Range of 79000 feet.

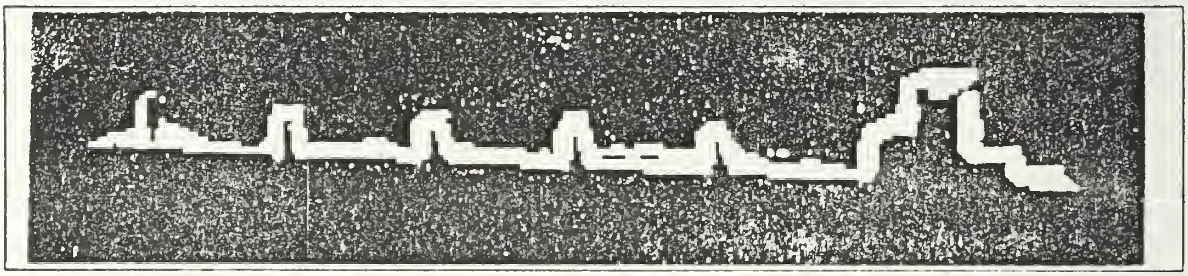


Figure 5.2 Before Closing Process.

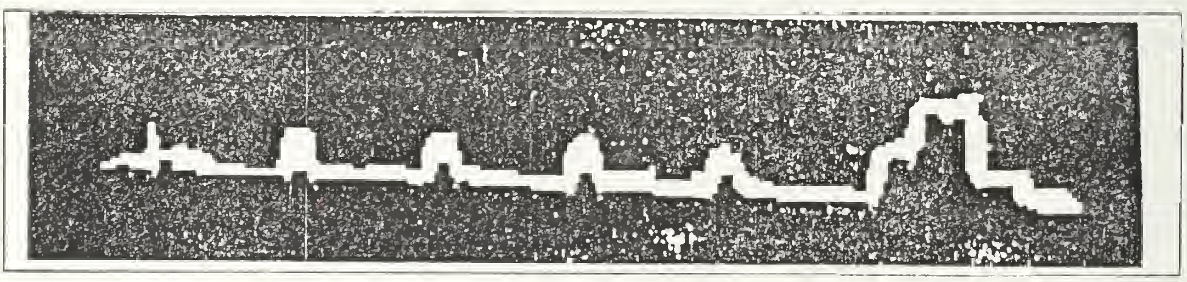


Figure 5.3 After Closing Process.

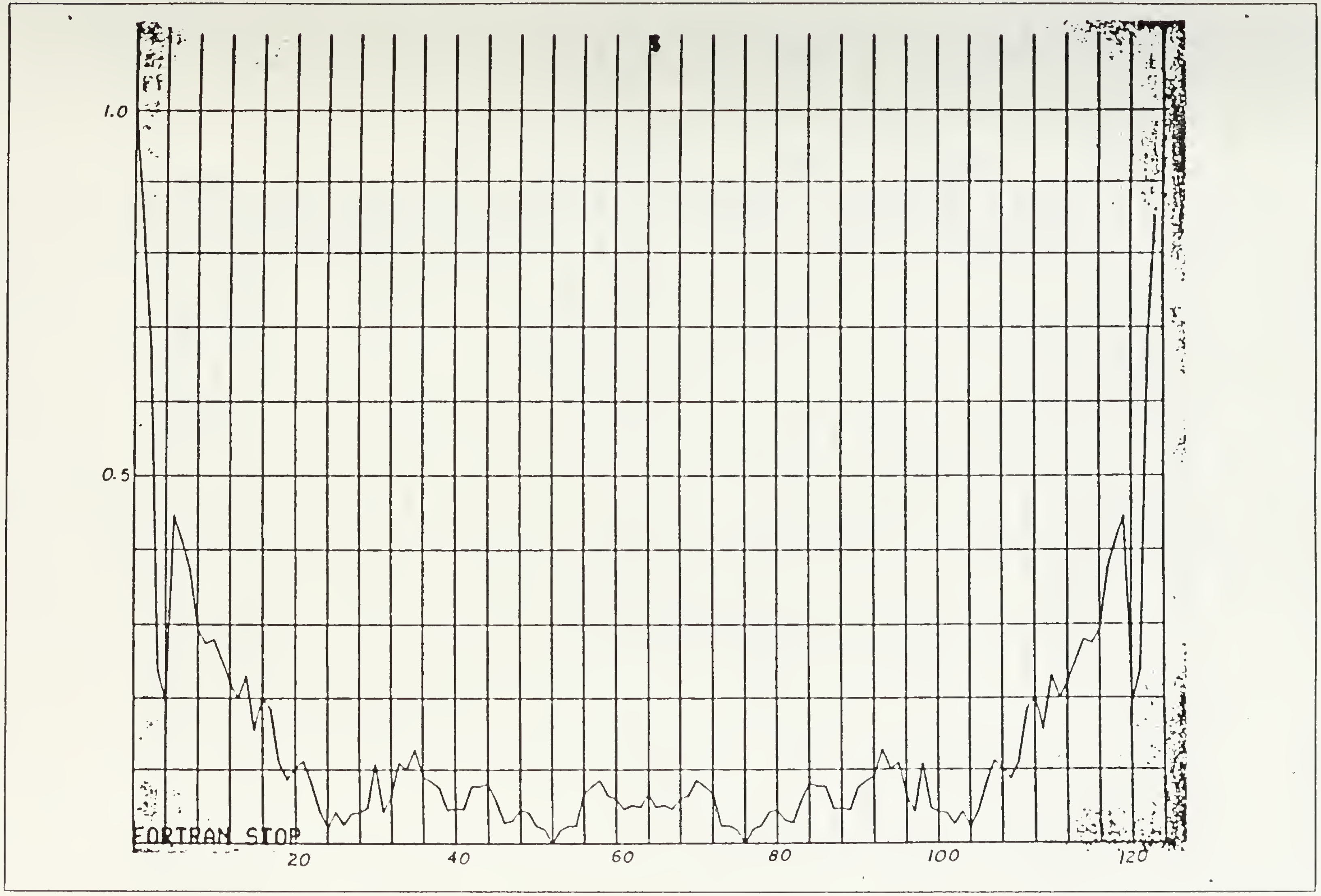


Figure 5.4   Logarithmic Magnitude of a CGN at a Range of 45000 feet.

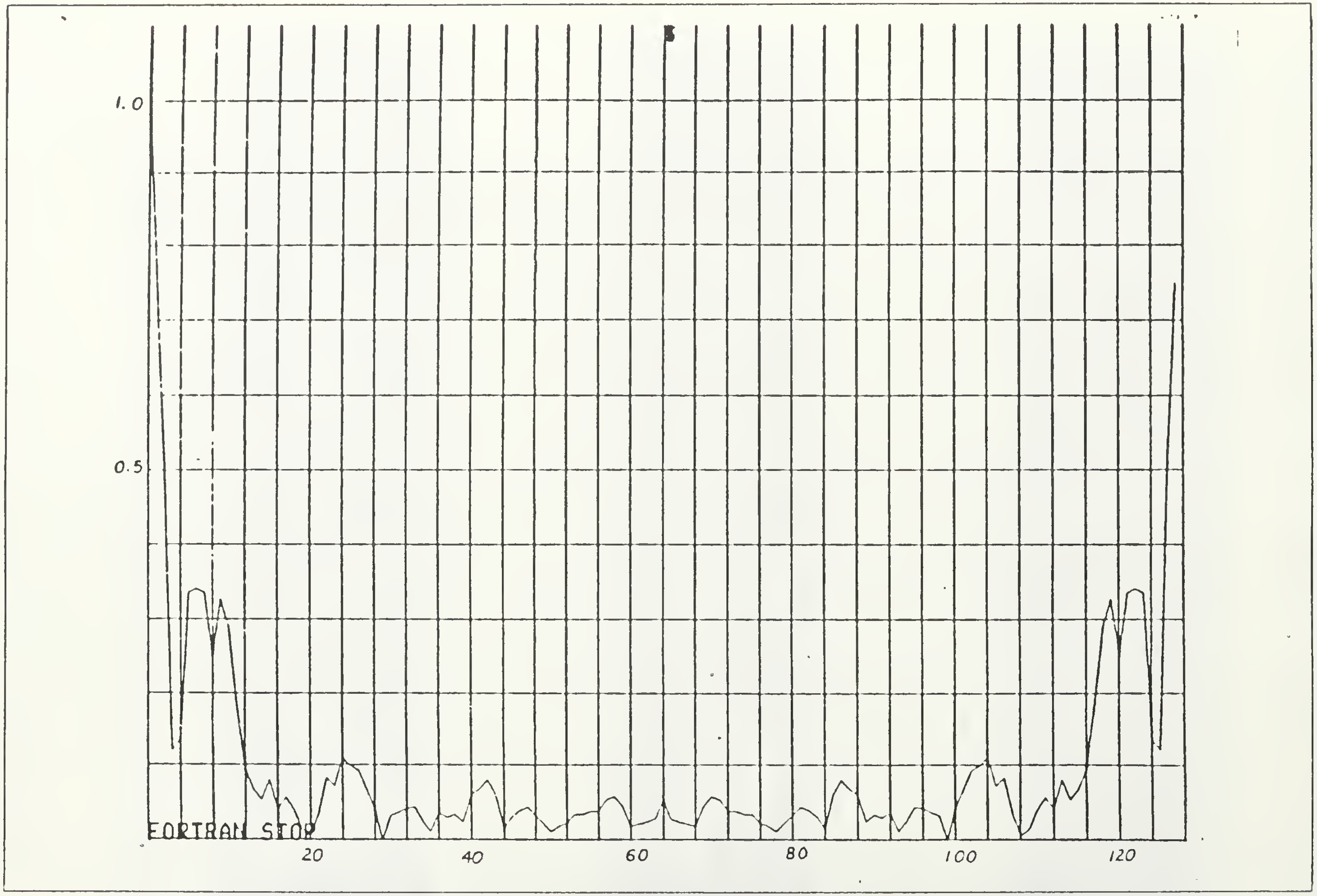


Figure 5.5 Logarithmic Magnitude of a CGN at a Range of 55000 feet.

In the B-spline coefficient method the technique of uneven knot selection has been employed. With the original sampling points on the ship profile as input, a smaller set of approximation samples are collected. These can be used to reconstruct the ship profile with sufficient information retained from the original image. The execution time as approached to that of handling the original data directly has been greatly reduced. The reduction of the number of sample points by almost a factor of 10 is common. For a CGN ship a set of original sampling points of 290, has been reduced to 36.

Comparing the two methods, Fourier Coefficient and B-spline Coefficient, the former method, in some cases is not effective to establish satisfactory classification of ships. This is due to difficulties in matching similarities of the shape of the coefficient curves. The latter, however, surpasses the former in that it is able to classify more ships accurately using computer programs. It is possible to improve the reliability of those two methods by reducing noise in the data collection process and the preprocessing process.

APPENDIX A

THE PROGRAM TO OBTAIN THE SUPERSTRUCTURE

```
program cut(input,output,infile,outfile);

  type
    byte = 0..255;
    imagerow1 = packed array [0..257] of byte;
    imagerow2 = packed array [0..255] of integer;
    imagerow3 = packed array [0..255] of byte;
    row1  = packed array [0..255] of byte;
    row2  = packed array [0..255] of integer;

  var
    sobel : array [0..63] of imagerow2;
    i,j  :  byte;
    f : array [0..65] of imagerow1;
    outfile : file of imagerow3;
    dx,dy,range,bright,n,m :integer;
    infile : file of row1;
    slope : real;
    image :array [0..63] of row1;
    x,x1,x2,y1,y2 : integer;
    NUM1,NUM2,NUM3,NUM4,max,min,max1 : integer;
    cal :array [0..63] of row1;
    thes : integer;
    NAME : PACKED ARRAY [1..20] OF CHAR;

    BEGIN

      WRITELN('INPUT SHIP  FILENAME OUT CUT.DAT');
      READLN(NAME);
      WRITELN('THRESHOLD');
      READ(THES);
      WRITELN('NUM TO CUT FROM LEFT');
      READ(NUM1);
      WRITELN('NUM2 TO CUT FROM RIGHT');
      READ(NUM2);
      WRITELN('NUM3 TO CUT FROM TOP');
      READ(NUM3);
      WRITELN('NUM4 TO CUT FROM BUTTOM');
      READ(NUM4);
      open (infile,NAME,history :=old,
            access_method :=sequential,
            record_length :=256,record_type :=fixed);
      open (outfile,'CUT.dat',history :=new,record_length :=256,
            record_type :=fixed);
      reset(infile);
      rewrite (outfile);
      i:=0;
      while not eof (infile) do
      begin
        read (infile,image[i]);
        for j:=0 to 255 do
          f[i+1,j+1] := image[i,j];
        i:=i+1;
      end;

 (*compute sobel*)
```

```
      FOR J:=0 TO 7 DO
        F[64,J+248]:=F[63,J+248];
      for i := 0 to 63 do begin
        f[i+1,0] := f[i+1,1];
        f[i+1,257] :=f[i+1,256];
      end;
      for j:= 0 to 255 do begin
        f[0,j+1]  := f[1,j+1];
        f[65,j+1] := f[64,j+1];
      end;
      f[0,0]:=f[1,1];
      f[0,257]:=f[1,256];
      f[65,0]:=f[64,1];
      f[65,257]:=f[64,256];

(*set initial max, min *)

      max:=0;
      max1:=0;
      min:=255;
      for i:= 0 to 63 do begin
        for j:= 0 to 255 do begin
          dx:=f[i,j]+2*f[i+1,j]+f[i+2,j]-f[i,j+2]
              -2*f[i+1,j+2]-f[i+2,j+2];
          dy:=f[i+2,j]+2*f[i+2,j+1]+f[i+2,j+2]-f[i,j]
              -2*f[i,j+1]-f[i,j+2];
          sobel[i,j]:=round((dx**2+dy**2)**0.5);
          if max < sobel[i,j] then
            max := sobel[i,j];
          if min > sobel[i,j] then
            min := sobel[i,j];
        end;
      end;
      range := max-min;

(* rescale*)

      for i:= 0 to 63 do begin
        for j:= 0 to 255 do begin
          cal[i,j] := round(((sobel[i,j]-min)*255)/range);
          if (j<=NUM1) or (j>=255-NUM2) then
            cal[i,j] :=0;
          if (i<=NUM3) or (i>=63-NUM4) then
            cal[i,j] := 0;

(* cut theshold*)
          IF THES<>0 THEN BEGIN
            if cal[i,j] <=thes
            then
              cal[i,j] := 0
            else
              cal[i,j] := 255;
          END;
        end;
      end;

(*find point at raw and aft*)
```

```
    m := 0;
    n := 0;
    bright:= 255;
    for j := 0 to 150 do begin
      for i:=0 to 63 do begin
        if(cal[i,j]=bright) and (n=0) then
        begin
          y1 := j;
          x1 := i;
          n := n+1;
        end; (*if*)
        if(cal[i,255-j]=bright) and (m=0) then
        begin
          y2 := 255-j;
          x2 := i;
          m := m+1;
        end; (*if*)
      end;(*for*)
    end;(*for*)
    writeln('x1=',x1,'y1=',y1,'x2=',x2,'y2=',y2);

(* cut line*)

    slope :=1.0;
    if x1=x2 then
    begin
      slope := 0;
      for i:=x1+1 to 63 do begin
        for j:=y1-5 to y2+5 do
          cal[i,j] :=0;
      end;(*for*)
    end;(*if*)
    if slope<>0 then
    begin
      slope :=(x2-x1)/(y2-y1);
      if slope < 0 then begin
        slope := abs(slope);
        for j := y1 to y2 do begin
          x := x1+1-round((j-y1)*slope);
          for i :=x to 63 do
            cal[i,j]:=0;
        end;(*for*)
      end;(*if*)
      if slope >0 then begin
        for j:= y1 to y2 do begin
          x := x1+1+round((j-y1)*slope);
          for i :=x to 63 do
            cal[i,j] :=0;
        end;(*for*)
      end;(*if*)
      writeln('step2');
    end;(*if*)

(*put outfile*)

    for i:=0 to 63 do begin
```

```
    for j :=0 to 255 do
      outfile^[j] := cal[i,j];
    put (outfile);
  end;(*for*)
  close(infile);
end.
```

APPENDIX B

THE PROGRAM OF CONTOUR FOLLOWING

```
PROGRAM TRACE(INPUT,OUTPUT,INFILE,OUTFILE);

  TYPE
    BYTE = 0..255;
    IMAGEROW1 = PACKED ARRAY [0..255] OF BYTE;
    ROW1 = PACKED ARRAY [0..257] OF BYTE;
  VAR
    R,C,F0,F1,F2,F3,F4,F5,F6,F7,F8  :  BYTE;
    F : ARRAY [0..65] OF ROW1;
    A,IMAGE : ARRAY [0..63] OF IMAGEROW1;
    INFILE : FILE OF IMAGEROW1;
    R1,C1,R2,C2,CDIR,I,J,COUNT : INTEGER;
    ROW,COL : ARRAY [0..512] OF INTEGER;
    OUTFILE : FILE OF IMAGEROW1;
    NAME : PACKED ARRAY [1..20] OF CHAR;

  PROCEDURE STORE;

  BEGIN
    COL[I]:=C-1;
    ROW[I]:=R-1;
    WRITELN('COL=',COL[I],'ROW=',ROW[I]);
    COUNT :=I;
    I:=I+1;
  END;

  PROCEDURE CMOVE;

  BEGIN
    F0 :=F[R-1,C]; F1:=F[R-1,C+1]; F2:=F[R,C+1]; F3:=F[R+1,C+1];
    F4:=F[R+1,C]; F5:=F[R+1,C-1]; F6:=F[R,C-1]; F7:=F[R-1,C-1];
    CASE CDIR OF
      0: BEGIN
        IF (F3=0) AND (F2=255) THEN BEGIN C:=C+1; CDIR:=1; END
        ELSE IF (F2=0) AND (F1=255) THEN BEGIN R:=R-1;
          STORE; C:=C+1; CDIR:=0; END
        ELSE IF (F1=0) AND (F0=255) THEN BEGIN R:=R-1;
          CDIR:=0; END
        ELSE IF (F0=0) AND (F7=255) THEN BEGIN C:=C-1;
          STORE; R:=R-1; CDIR:=3; END
        ELSE IF (F7=0) AND (F6=255) THEN BEGIN C:=C-1;
          CDIR:=3; END
        ELSE IF (F6=0) AND (F5=255) THEN BEGIN R:=R+1;
          STORE; C:=C-1; CDIR:=3; END
        ELSE BEGIN R:=R+1; CDIR:=2; END;
      END;
      1: BEGIN
        IF (F5=0) AND (F4=255) THEN BEGIN R:=R+1;
          CDIR:=2; END
        ELSE IF (F4=0) AND (F3=255) THEN BEGIN C:=C+1;
          STORE; R:=R+1; CDIR:=1; END
        ELSE IF (F3=0) AND (F2=255) THEN BEGIN C:=C+1;
          CDIR:=1; END
        ELSE IF (F2=0) AND (F1=255) THEN BEGIN R:=R-1;
          STORE; C:=C+1; CDIR:=0; END
        ELSE IF (F1=0) AND (F0=255) THEN BEGIN R:=R-1;
          CDIR:=0; END
```

```
      ELSE IF (F0=0) AND (F7=255) THEN BEGIN C:=C-1;
        STORE; R:=R-1; CDIR:=0; END
      ELSE BEGIN C:=C-1; CDIR:=3; END;
    END;
    2: BEGIN
      IF (F7=0) AND (F6=255) THEN BEGIN C:=C-1;
        CDIR:=3; END
      ELSE IF (F6=0) AND (F5=255) THEN BEGIN R:=R+1;
        STORE; C:=C-1; CDIR:=3; END
      ELSE IF (F5=0) AND (F4=255) THEN BEGIN R:=R+1;
        CDIR:=2; END
      ELSE IF (F4=0) AND (F3=255) THEN BEGIN C:=C+1;
        STORE; R:=R+1; CDIR:=1; END
      ELSE IF (F3=0) AND (F2=255) THEN BEGIN C:=C+1;
        CDIR:=1; END
      ELSE IF (F2=0) AND (F1=255) THEN BEGIN R:=R-1;
        STORE; C:=C+1; CDIR:=1; END
      ELSE BEGIN R:=R-1; CDIR:=0; END;
    END;
    3: BEGIN
      IF (F1=0) AND (F0=255) THEN BEGIN R:=R-1;
        CDIR:=0; END
      ELSE IF (F0=0) AND (F7=255) THEN BEGIN C:=C-1;
        STORE; R:=R-1; CDIR:=0; END
      ELSE IF (F7=0) AND (F6=255) THEN BEGIN C:=C-1;
        CDIR:=3; END
      ELSE IF (F6=0) AND (F5=255) THEN BEGIN R:=R+1;
        STORE; C:=C-1; CDIR:=2; END
      ELSE IF (F5=0) AND (F4=255) THEN BEGIN R:=R+1;
        CDIR:=2; END
      ELSE IF (F4=0) AND (F3=255) THEN BEGIN C:=C+1;
        STORE; R:=R+1; CDIR:=2; END
      ELSE BEGIN C:=C+1; CDIR:=1; END;
    END;
  END;
END;


PROCEDURE INITIAL;

BEGIN
  FOR I :=0 TO 255 DO BEGIN
    COL[I] :=0;
    ROW[I] :=0;
  END;
  FOR J:=0 TO 63 DO BEGIN
    FOR I:=0 TO 255 DO
      A[R,C]:=0;
  END;
  I:=0;
END;

PROCEDURE FIRST;

  VAR
    N,M :INTEGER;
BEGIN
```

```
    N := 0; M:=0;
    FOR C :=0 TO 200 DO BEGIN
      FOR R :=0 TO 63 DO BEGIN
        IF (F[R,C]=255) AND (N=0) THEN BEGIN
          C1 := C; R1 :=R; N := N+1;
        END;
        IF (F[R,255-C]=255) AND (M=0) THEN BEGIN
          C2 := 255-C; R2 := R; M :=M+1;
        END;
      END;
    END;
    WRITELN('COL1=',C1,'ROW1=',R1,'COL2=',C2,'ROW2=',R2);
  END;

(************************************)
(* MAIN PROGRAM *)
  BEGIN
    WRITELN('INPUT CUT OR CONLINE FILENAME');
    READLN(NAME);
    OPEN (INFILE,NAME,HISTORY := OLD,
      ACCESS_METHOD :=SEQUENTIAL,
      RECORD_LENGTH :=256,RECORD_TYPE :=FIXED);
    OPEN (OUTFILE,'TR.DAT',HISTORY :=NEW,RECORD_LENGTH :=256,
      RECORD_TYPE :=FIXED);
    RESET(INFILE);
    REWRITE (OUTFILE);
    R :=0;
    WHILE NOT EOF (INFILE) DO
    BEGIN
      READ (INFILE,IMAGE[R]);
      FOR C := 0 TO 255 DO
        F[R+1,C+1] := IMAGE[R,C];
      R := R+1;
    END;
    FIRST;
    R :=R1; C :=C1; COIR :=1;
    INITIAL;
    WHILE (C<>C2) DO BEGIN
      STORE;
      CMOVE;
    END;
    FOR I:=0 TO COUNT DO
      A[ROW[I],COL[I]] :=255;
    FOR R:=0 TO 63 DO BEGIN
      FOR C:=0 TO 255 DO
        OUTFILE^[C] := A[R,C];
    PUT(OUTFILE);
  END;
  WRITELN('NUMBER=',COUNT);
  CLOSE(INFILE);
END.
```

APPENDIX C

THE SUBROUTINE "PARAM"

```
      SUBROUTINE PARAM(X,Y,Z,W,M,ZB,ZE,K,S,N,T,CX,CY,FP,IOPT,IPAR,IER)
C  GIVEN THE SET OF DATA POINTS (X(I),Y(I)) WITH CORRESPONDING Z-
C  VALUES Z(I),I=1,2,...M AND GIVEN ALSO THE SET OF POSITIVE
C  NUMBERS W(I),I=1,2,...M, SUBROUTINE PARAM FINDS A SMOOTH APPROXIMAT-
C  ING CURVE WITH PARAMETER REPRESENTATION X = SX(Z), Y = SY(Z).
C  SX(Z) AND SY(Z) ARE TWO SPLINE FUNCTIONS OF DEGREE K WITH THE NUMBER
C  AND THE POSITION OF THE KNOTS T(J),J=1,2,...N AUTOMATICALLY
C  CHOSEN BY THE ROUTINE. THE SMOOTHNESS OF SX(Z) AND SY(Z) IS
C  ACHIEVED BY MINIMALIZING THE SUM(DX(R)**2+DY(R)**2) WHERE DX(R)
C  AND DY(R) STAND FOR THE DISCONTINUITYJUMP OF THE KTH DERIVATIVE
C  OF SX(Z) AND SY(Z) AT THE KNOT T(R), R=K+2,...N-K-1.
C  THE AMOUNT OF SMOOTHNESS IS DETERMINED BY THE CONDITION THAT F(P) =
C  SUM(W(I)*((X(I)-SX(Z(I)))**2 +(Y(I)-SY(Z(I)))**2)) BE <= S, WITH
C  S A GIVEN NON-NEGATIVE CONSTANT.
C  THE SPLINE FUNCTIONS SX(Z) AND SY(Z) ARE GIVEN IN THEIR B-SPLINE
C  REPRESENTATION (B-SPLINE COEFFICIENTS CX(J),RESP. CY(J),J=1,...N-K-1)
C  AND CAN BE EVALUATED BY MEANS OF FUNCTION DERIV.
C  CALLING SEQUENCE:
C     CALL PARAM(X,Y,Z,W,M,ZB,ZE,K,S,N,T,CX,CY,FP,IOPT,IPAR,IER)
C
C  INPUT PARAMETERS:
C    X    : ARRAY,LENGTH M, CONTAINING THE ABSCISSAE OF THE DATA POINTS
C    Y    : ARRAY,LENGTH M, CONTAINING THE ORDINATES OF THE DATA POINTS
C    W    : ARRAY,MINIMUM LENGTH M,CONTAINING THE WEIGHTS W(I).
C    M    : INTEGER VALUE,CONTAINING THE NUMBER OF DATA POINTS.
C    K    : INTEGER VALUE,CONTAINING THE DEGREE OF SX(Z) AND SY(Z).
C    S    : REAL VALUE,CONTAINING THE SMOOTHING FACTOR.
C    IOPT : INTEGER VALUE WHICH TAKES THE VALUE 0 OR 1.
C      IOPT=0: THE ROUTINE WILL RESTART ALL COMPUTATIONS.
C      IOPT=1: THE ROUTINE WILL START WITH THE KNOTS FOUND AT THE
C              LAST CALL OF THE ROUTINE. IF IOPT=1 THE OUTPUT
C              PARAMETERS T AND N ARE INPUT PARAMETERS AS WELL.
C              IF IOPT=1 THE USER MUST PROVIDE WITH A COMMON BLOCK
C                 COMMON/OPT1/NRDATA(NEST),FP0,FPOLD,NPLUS
C    IPAR : INTEGER FLAG.
C      IPAR = 0: FOR EACH DATA POINT (X(I),Y(I)) THE PROGRAM AUTOMATICALLY
C                CHOOSES A CORRESPONDING VALUE OF THE PARAMETER Z, I.E.
C                Z(1)=0:Z(I)=Z(I-1)+SQRT((X(I)-X(I-1))**2+(Y(I)-Y(I-1))**2)
C                THE BOUNDARIES FOR THE PARAMETER Z ARE CHOSEN AS FOLLOWS
C                  ZB = Z(1) : ZE = Z(M).
C      IPAR = 1: THE USER HIMSELF PROVIDES WITH THE VALUES OF THE
C                PARAMETER Z AND WITH THE BOUNDARIES ZB AND ZE.
C    Z    : ARRAY,LENGTH M, CONTAINING THE VALUES OF THE PARAMETER Z
C           (IPAR = 1)
C    ZB,ZE: REAL VALUES, CONTAINING THE BOUNDARIES OF THE PARAMETER Z
C           (IPAR = 1).
C
C  OUTPUT PARAMETERS:
C    T    : ARRAY,LENGTH NEST (SEE DATA INITIALIZATION STATEMENT),
C           WHICH CONTAINS THE POSITION OF THE KNOTS,I.E. THE POSITION
C           OF THE INTERIOR KNOTS T(K+2),...T(N-K-1), AS WELL AS THE
C           POSITION OF THE KNOTS T(1)=T(2)=...=T(K+1)=ZB AND ZE =
C           T(N-K)=...=T(N) WHICH ARE NEEDED FOR THE B-SPLINE REPRESENT.
C    CX,CY: ARRAYS,LENGTH NEST, CONTAINING THE B-SPLINE COEFFICIENTS
C           OF SX(Z), RESP. SY(Z).
C    N    : INTEGER VALUE,CONTAINING THE TOTAL NUMBER OF KNOTS.
C    FP   : REAL VALUE, WHICH CONTAINS THE SUM(WI*(XI-SX(ZI))**2)
C           + SUM(WI*(YI-SY(ZI))**2), I=1,2,...M.
C    IER  : ERROR CODE
C      IER=0: NORMAL RETURN.
C      IER=-1:NORMAL RETURN,SX(Z) AND SY(Z) ARE INTERPOLATING SPLINES
```

```
C       IER=-2:NORMAL RETURN, SX(Z) AND SY(Z) ARE POLYNOMIALS OF DEGREE K
C       IER>0 :ABNORMAL TERMINATION.
C         IER=1:THE REQUIRED STORAGE SPACE EXCEEDS THE AVAILABLE
C               STORAGE SPACE,SPECIFIED BY THE PARAMETER NEST.
C               PROBABLY CAUSES:NEST OR S TOO SMALL.
C         IER=2:A THEORETICALLY IMPOSSIBLE RESULT WAS FOUND DURING
C               THE ITERATION PROCESS.
C               PROBABLY CAUSES:TOL TOO SMALL
C         IER=3:THE MAXIMAL NUMBER OF ITERATIONS MAXIT HAS BEEN
C               REACHED.
C               PROBABLY CAUSES:MAXIT OR TOL TOO SMALL.
C         IER=10:SOME OF THE INPUT DATA ARE INVALID(SEE RESTRICTIONS).
C
C  RESTRICTIONS:
C    1) M > K > 0
C    2) ZB <= Z(R) < Z(R+1) <= ZE, R=1,2,...M-1.  (IPAR = 1)
C    3) W(R) > 0, R=1,2,...M.
C    4) S >= 0.
C    5) NEST >= 2*K+2.
C
C  OTHER SUBROUTINES REQUIRED:
C    BSPLIN,COSSIN,ROTATE,BACK,NKNOT,DISCO AND RATION.
      DIMENSION X(M),Y(M),W(M),Z(M),T(200),CX(200),CY(200),
     < FPINT(200),RX(200),RY(200),DIAG(200),DPRIME(200),
     < G(200,6),B(200,7),Q(400,6),H(7),NRDATA(200),A(200,5)
C  COMMON/OPT1/NRDATA(NEST),FP0,FPOLD,NPLUS
C    NRDATA: INTEGER ARRAY,LENGTH NEST,WHICH GIVES THE NUMBER OF
C            DATA POINTS INSIDE EACH KNOT INTERVAL.
C    FP0   : REAL VALUE, WHICH CONTAINS THE SUM(WI*(XI-SX(ZI))**2)+
C            SUM(WI*(YI-SY(ZI))**2) WITH SX(Z) AND SY(Z) LEAST-SQUARES
C            POLYNOMIALS OF DEGREE K.
C    FPOLD : REAL VALUE,WHICH CONTAINS THE SUM(WI*(XI-SX(ZI))**2)+
C            SUM(WI*(YI-SY(ZI))**2) WITH SX(Z) AND SY(Z) LEAST-SQUARES
C            SPLINE FUNCTIONS CORRESPONDING TO THE LAST FOUND SET OF
C            KNOTS BUT ONE.
C    NPLUS : INTEGER VALUE,GIVING THE NUMBER OF KNOTS OF THE LAST
C            SET MINUS THE NUMBER OF THE LAST SET BUT ONE.
      COMMON/OPT1/NRDATA,FP0,FPOLD,NPLUS
C  DATA INITIALIZATION STATEMENT TO SPECIFY
C    TOL  : THE REQUESTED RELATIVE ACCURACY FOR THE ROOT OF F(P) = S.
C    MAXIT: THE MAXIMAL NUMBER OF ITERATIONS ALLOWED.
C    NEST : AN OVER-ESTIMATE OF THE NUMBER OF KNOTS N. THIS PARAMETER
C          MUST BE SET BY THE USER TO INDICATE THE STORAGE SPACE
C          AVAILABLE TO THE SUBROUTINE. THE DIMENSION SPECIFICATIONS
C          OF THE ARRAYS  T,CX,CY,NRDATA,FPINT,RX,RY,DIAG,DPRIME(N),
C          A(N,K),G(N,K+1),B(N,K+2),Q(M,K+1) AND H(K+2) DEPEND
C          ON N,M AND K. SINCE N IS UNKNOWN AT THE TIME THE
C          USER SETS UP THE DIMENSION INFORMATION AN OVER-ESTIMATE
C          OF THESE ARRAYS WILL GENERALLY BE MADE. THE FOLLOWING
C          REMARKS ARE INTENDED TO HELP THE USER
C            1) 2*K+2 <= N <= M+K+1
C            2) THE SMALLER THE VALUE OF S, THE GREATER N WILL BE.
C            3) NORMALLY N = M/2 IS AN OVER-ESTIMATE.
      DATA TOL/0.001/,MAXIT/20/,NEST/200/
C  BEFORE STARTING COMPUTATIONS A DATA CHECK IS MADE. IF THE INPUT
C  DATA ARE INVALID CONTROLE IS IMMEDIATELY REPASSED TO THE DRIVER
C  PROGRAM (IER=10).
      IER = 0
      K1 = K+1
      K2 = K1+1
      NMIN = 2*K1
```

```
      IF(K.LE.0) IER = 10
      IF(M.LT.K1 .OR. NEST.LT.NMIN) IER = 10
      IF(S.LT.0.) IER = 10
      IF(IER.NE.0) GO TO 440
C  CHECK WHETHER THE Z-VALUES ARE PROVIDED WITH BY THE USER.
      IF(IPAR.NE.0) GO TO 6
C  FIND FOR EACH DATA POINT A CORRESPONDING VALUE OF THE PARAMETER Z
C  AND FIX THE BOUNDARIES ZB AND ZE.
      Z(1) = 0.
      DO 4 I=2,M
        Z(I) = Z(I-1)+SQRT((X(I)-X(I-1))**2+(Y(I)-Y(I-1))**2)
   4  CONTINUE
      ZB = Z(1)
      ZE = Z(M)
   6  IF(ZB.GT.Z(1) .OR. ZE.LT.Z(M) .OR. W(1).LE.0.) IER = 10
      DO 10 I=2,M
        IF(Z(I-1).GE.Z(I) .OR. W(I).LE.0.) IER = 10
  10  CONTINUE
      IF(IER.NE.0) GO TO 440
C  CALCULATION OF ACC, THE ABSOLUTE TOLERANCE FOR THE ROOT OF F(P)=S.
      ACC = TOL*S
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  PART 1: DETERMINATION OF THE NUMBER OF KNOTS AND THEIR POSITION     C
C  **************************************************************      C
C  GIVEN A SET OF KNOTS WE COMPUTE THE LEAST-SQUARES SPLINES SXINF(Z)  C
C  AND SYINF(Z).IF THE SUM F(P=INF)<=S WE ACCEPT THE CHOICE OF KNOTS.  C
C  OTHERWISE WE HAVE TO INCREASE THEIR NUMBER.                         C
C  THE INITIAL CHOICE OF KNOTS DEPENDS ON THE VALUE OF S AND IOPT.     C
C    IF S=0 WE HAVE SPLINE INTERPOLATION; IN THAT CASE THE NUMBER OF   C
C    KNOTS EQUALS NMAX = M+K+1.                                        C
C    IF S > 0 AND                                                      C
C      IOPT=0 WE FIRST COMPUTE THE LEAST-SQUARES POLYNOMIALS OF        C
C      DEGREE K; N = NMIN = 2*K+2                                      C
C      IOPT=1 WE START WITH THE SET OF KNOTS FOUND AT THE LAST         C
C      CALL OF THE ROUTINE, EXCEPT FOR THE CASE THAT S > FP0; THEN     C
C      WE COMPUTE DIRECTLY THE LEAST-SQUARES POLYNOMIALS OF DEGREE K.  C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  DETERMINE NMAX, THE NUMBER OF KNOTS FOR SPLINE INTERPOLATION.
      NMAX = M+K1
      IF(S.GT.0.) GO TO 45
C  IF S=0, SX(Z) AND SY(Z) ARE INTERPOLATING SPLINES.
      N = NMAX
C  TEST WHETHER THE REQUIRED STORAGE SPACE EXCEEDS THE AVAILABLE ONE.
      IF(N.GT.NEST) GO TO 420
C  FIND THE POSITION OF THE INTERIOR KNOTS IN CASE OF INTERPOLATION.
      MK1 = M-K1
      IF(MK1.EQ.0) GO TO 60
      K3 = K/2
      I = K2
      J = K3+2
      IF(K3*2.EQ.K) GO TO 30
      DO 20 L=1,MK1
        T(I) = Z(J)
        I = I+1
        J = J+1
  20  CONTINUE
      GO TO 60
  30  DO 40 L=1,MK1
        T(I) = (Z(J)-Z(J-1))*0.5
        I = I+1
        J = J+1
```

```
 40   CONTINUE
      GO TO 60
C  IF S>0 OUR INITIAL CHOICE OF KNOTS DEPENDS ON THE VALUE OF IOPT.
C  IF IOPT=0 OR IOPT=1 AND S>=FP0, WE START COMPUTING THE LEAST-SQUARES
C  POLYNOMIALS OF DEGREE K WHICH ARE SPLINES WITHOUT INTERIOR KNOTS.
C  IF IOPT=1 AND FP0>S WE START COMPUTING THE LEAST-SQUARES SPLINES
C  ACCORDING TO THE SET OF KNOTS FOUND AT THE LAST CALL OF THE ROUTINE.
 45   IF(IOPT.LE.0) GO TO 50
      IF(FP0.GT.S) GO TO 60
 50   N = NMIN
      NRDATA(1) = M-2
C  MAIN LOOP FOR THE DIFFERENT SETS OF KNOTS. M IS A SAVE UPPER BOUND
C  FOR THE NUMBER OF TRIALS.
 60   DO 200 ITER = 1,M
        IF(N.EQ.NMIN) IER = -2
C  FIND NRINT, THE NUMBER OF KNOT INTERVALS.
        NRINT = N-NMIN+1
C  FIND THE POSITION OF THE ADDITIONAL KNOTS WHICH ARE NEEDED FOR
C  THE B-SPLINE REPRESENTATION OF SX(Z) AND SY(Z).
        NK1 = N-K1
        I = N
        DO 70 J=1,K1
          T(J) = ZB
          T(I) = ZE
          I = I-1
 70     CONTINUE
C  COMPUTE THE B-SPLINE COEFFICIENTS OF THE LEAST-SQUARES SPLINES SXINF(Z)
C  AND SYINF(Z). THE OBSERVATION MATRIX A IS BUILT UP ROW BY ROW AND
C  REDUCED TO UPPER TRIANGULAR FORM BY GIVENS TRANSFORMATIONS
C  WITHOUT SQUARE ROOTS. AT THE SAME TIME FP=F(P=INF) IS COMPUTED
        FP = 0.
C  INITIALIZE THE OBSERVATION MATRIX A.
        DO 80 I=1,NK1
          DIAG(I) = 0.
          RX(I) = 0.
          RY(I) = 0.
          DO 80 J=1,K
            A(I,J) = 0.
 80     CONTINUE
        L = K1
        DO 130 IT=1,M
C  FETCH THE CURRENT DATA POINT X(IT),Y(IT),Z(IT).
          XI = X(IT)
          YI = Y(IT)
          ZI = Z(IT)
          WI = W(IT)
C  SEARCH FOR KNOT INTERVAL T(L) <= ZI <= T(L+1).
          IF(ZI.GE.T(L+1) .AND. L.NE.NK1) L = L+1
C  EVALUATE THE (K+1) NON-ZERO B-SPLINES AT ZI AND STORE THEM IN Q.
          CALL BSPLIN(T,N,K,ZI,L,H)
          DO 90 I=1,K1
            IF(H(I).LT.0.1E-07) H(I) = 0.
            Q(IT,I) = H(I)
 90       CONTINUE
C  ROTATE THE NEW ROW OF THE OBSERVATION MATRIX INTO TRIANGLE BY
C  GIVENS TRANSFORMATIONS WITHOUT SQUARE ROOTS.
          J = L-K1
          DO 110 I=1,K1
            IF(WI.EQ.0.) GO TO 130
            J = J+1
            PIV = H(I)
```

```
            IF(PIV.EQ.0.) GO TO 110
C  CALCULATE THE PARAMETERS OF THE GIVENS TRANSFORMATION.
            CALL COSSIN(PIV,WI,DIAG(J),COS,SIN)
C  TRANSFORMATIONS TO RIGHT HAND SIDES.
            CALL ROTATE(PIV,COS,SIN,XI,RX(J))
            CALL ROTATE(PIV,COS,SIN,YI,RY(J))
            IF(I.EQ.K1) GO TO 120
            I2 = 0
            I3 = I+1
            DO 100 I1 = I3,K1
              I2 = I2+1
C  TRANSFORMATIONS TO LEFT HAND SIDE.
              CALL ROTATE(PIV,COS,SIN,H(I1),A(J,I2))
 100        CONTINUE
 110      CONTINUE
C  ADD CONTRIBUTION OF THIS ROW TO THE SUM OF SQUARES OF RESIDUAL
C  RIGHT HAND SIDES.
 120      FP = FP+WI*(XI**2+YI**2)
 130    CONTINUE
        IF(IER.EQ.-2) FP0 = FP
C  BACKWARD SUBSTITUTION TO OBTAIN THE B-SPLINE COEFFICIENTS.
        CALL BACK(A,RX,NK1,K,CX)
        CALL BACK(A,RY,NK1,K,CY)
C  TEST WHETHER THE APPROXIMATION X=SXINF(Z),Y=SYINF(Z) IS AN
C  ACCEPTABLE SOLUTION.
        FPMS = FP-S
        IF(ABS(FPMS).LT.ACC) GO TO 440
C  IF F(P=INF) < S ACCEPT THE CHOICE OF KNOTS.
        IF(FPMS.LT.0.) GO TO 250
C  IF N=NMAX,SXINF(Z) AND SYINF(Z) ARE INTERPOLATING SPLINES.
        IF(N.EQ.NMAX) GO TO 430
C  INCREASE THE NUMBER OF KNOTS.
C  IF N=NEST WE CANNOT INCREASE THE NUMBER OF KNOTS BECAUSE OF
C  THE STORAGE CAPACITY LIMITATION.
        IF(N.EQ.NEST) GO TO 420
C  DETERMINE THE NUMBER OF KNOTS NPLUS WE ARE GOING TO ADD.
        IF(IER.EQ.0) GO TO 140
        NPLUS = 1
        IER = 0
        GO TO 150
 140    NPL1 = NPLUS*2
        IF(FPOLD-FP.GT.ACC) NPL1 = FLOAT(NPLUS)*FPMS/(FPOLD-FP)
        NPLUS = MIN0(NPLUS*2,MAX0(NPL1,NPLUS/2,1))
 150    FPOLD = FP
C  COMPUTE THE SUM(WI*((XI-SXINF(ZI))**2+(YI-SYINF(ZI))**2)) FOR
C  EACH KNOT INTERVAL T(J+K) <= ZI <= T(J+K+1) AND STORE IT IN
C  FPINT(J),J=1,2,...NRINT.
        FPART = 0.
        I = 1
        L = K2
        NEW = 0
        DO 180 IT=1,M
          IF(Z(IT).LT.T(L) .OR. L.GT.NK1) GO TO 160
          NEW = 1
          L = L+1
 160      TERM1 = 0.
          TERM2 = 0.
          L0 = L-K2
          DO 170 J=1,K1
            L0 = L0+1
            TERM1 = TERM1+CX(L0)*Q(IT,J)
```

```
            TERM2 = TERM2+CY(LO)*C(IT,J)
 170      CONTINUE
          TERM = W(IT)*((TERM1-X(IT))**2+(TERM2-Y(IT))**2)
          FPART = FPART+TERM
          IF(NEW.EQ.0) GO TO 180
          STORE = TERM*0.5
          FPINT(I) = FPART-STORE
          I = I+1
          FPART = STORE
          NEW = 0
 180    CONTINUE
        FPINT(NRINT) = FPART
        DO 190 L=1,NPLUS
C  ADD A NEW KNOT.
          CALL NKNOT(Z,M,T,N,FPINT,NRDATA,NRINT)
C  TEST WHETHER WE CANNOT FURTHER INCREASE THE NUMBER OF KNOTS.
          IF(N.EQ.NMAX .OR. N.EQ.NEST) GO TO 200
 190    CONTINUE
C  RESTART THE COMPUTATIONS WITH THE NEW SET OF KNOTS.
 200  CONTINUE
C  TEST WHETHER THE APPROXIMATION X=SX(Z),Y=SY(Z) WITH SX(Z) AND SY(Z)
C  THE LEAST-SQUARES KTH DEGREE POLYNOMIALS, IS A SOLUTION OF OUR PROBLEM
 250  IF(IER.EQ.-2) GO TO 440
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  PART 2: DETERMINATION OF THE SMOOTHING SPLINES SXP(Z) AND SYP(Z).  C
C  ***************************************************************   C
C  WE HAVE DETERMINED THE NUMBER OF KNOTS AND THEIR POSITION.         C
C  WE NOW COMPUTE THE B-SPLINE COEFFICIENTS OF THE SMOOTHING SPLINES  C
C  SXP(Z) AND SYP(Z). THE OBSERVATION MATRIX A IS EXTENDED BY THE ROWS C
C  OF MATRIX B EXPRESSING THAT THE KTH DERIVATIVE DISCONTINUITIES OF  C
C  SXP(Z) AND SYP(Z) AT THE INTERIOR KNOTS T(K+2),...T(N-K-1) MUST BE C
C  ZERO. THE CORRESPONDING WEIGHTS OF THESE ADDITIONAL ROWS ARE SET   C
C  TO 1/SQRT(P). ITERATIVELY WE THEN HAVE TO DETERMINE THE VALUE OF P C
C  SUCH THAT F(P)=SUM(WI*((XI-SXP(ZI))**2+(YI-SYP(ZI))**2) BE = S. WE C
C  ALREADY KNOW THAT THE LEAST-SQUARES POLYNOMIALS CORRESPOND TO P=0, C
C  AND THAT THE LEAST-SQUARES SPLINES CORRESPOND TO P=INFINITY. THE   C
C  ITERATION PROCESS WHICH IS PROPOSED HERE, MAKES USE OF RATIONAL    C
C  INTERPOLATION. SINCE F(P) IS A CONVEX AND STRICTLY DECREASING      C
C  FUNCTION OF P, IT CAN BE APPROXIMATED BY A RATIONAL FUNCTION       C
C  R(P) = (U*P+V)/(P+W). THREE VALUES OF P(P1,P2,P3) WITH CORRESPOND- C
C  ING VALUES OF F(P) (F1=F(P1)-S,F2=F(P2)-S,F3=F(P3)-S) ARE USED     C
C  TO CALCULATE THE NEW VALUE OF P SUCH THAT R(P)=S. CONVERGENCE IS   C
C  GUARANTEED BY TAKING F1>0 AND F3<0.                                C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  EVALUATE THE DISCONTINUITY JUMP OF THE KTH DERIVATIVE OF THE
C  B-SPLINES AT THE KNOTS T(L),L=K+2,...N-K-1 AND STORE IN B.
      CALL DISCO(T,N,K2,B)
C  INITIAL VALUE FOR P.
      P1 = 0.
      F1 = FP0-S
      P3 = -1.
      F3 = FPMS
      P = -F1/F3
      ICHECK = 0
      N8 = N-NMIN
C  ITERATION PROCESS TO FIND THE ROOT OF F(P) = S.
      DO 350 ITER=1,MAXIT
C  THE ROWS OF MATRIX B WITH WEIGHT 1/SQRT(P) ARE ROTATED INTO
C  THE TRIANGULARISED OBSERVATION MATRIX A WHICH IS STORED IN G.
        PINV = 1.0/P
        DO 260 I=1,NK1
```

```
          DPRIME(I) = DIAG(I)
          CX(I) = RX(I)
          CY(I) = RY(I)
          G(I,K1) = 0.
          DO 260 J=1,K
            G(I,J) = A(I,J)
 260    CONTINUE
        DO 300 IT=1,N8
C  THE ROW OF MATRIX B IS ROTATED INTO TRIANGLE BY GIVENS TRANSFORMATIONS.
          DO 270 I=1,K2
            H(I) = B(IT,I)
 270      CONTINUE
          XI = 0.
          YI = 0.
          WI = PINV
          DO 290 J=IT,NK1
            IF(WI.EQ.0.) GO TO 300
            PIV = H(1)
C  CALCULATE THE PARAMETERS OF THE GIVENS TRANSFORMATION.
            CALL COSSIN(PIV,WI,DPRIME(J),COS,SIN)
C  TRANSFORMATIONS TO RIGHT HAND SIDES.
            CALL ROTATE(PIV,COS,SIN,XI,CX(J))
            CALL ROTATE(PIV,COS,SIN,YI,CY(J))
            IF(J.EQ.NK1) GO TO 300
            I2 = K1
            IF(J.GT.N8) I2 = NK1-J
            DO 280 I=1,I2
C  TRANSFORMATIONS TO LEFT HAND SIDE.
              CALL ROTATE(PIV,COS,SIN,H(I+1),G(J,I))
              H(I) = H(I+1)
 280        CONTINUE
            H(I2+1) = 0.
 290      CONTINUE
 300    CONTINUE
C  BACKWARD SUBSTITUTION TO OBTAIN THE B-SPLINE COEFFICIENTS.
        CALL BACK(G,CX,NK1,K,CX)
        CALL BACK(G,CY,NK1,K,CY)
C  COMPUTATION OF F(P).
        FP = 0.
        L = K2
        DO 330 IT=1,M
          IF(Z(IT).LT.T(L) .OR. L.GT.NK1) GO TO 310
          L = L+1
 310      L0 = L-K2
          TERM1 = 0.
          TERM2 = 0.
          DO 320 J=1,K1
            L0 = L0+1
            TERM1 = TERM1+CX(L0)*Q(IT,J)
            TERM2 = TERM2+CY(L0)*Q(IT,J)
 320      CONTINUE
          FP = FP+W(IT)*((TERM1-X(IT))**2+(TERM2-Y(IT))**2)
 330    CONTINUE
C  TEST WHETHER THE APPROXIMATION X=SXP(Z),Y=SYP(Z) IS AN ACCEPTABLE
C  SOLUTION.
        FPMS = FP-S
        IF(ABS(FPMS).LT.ACC) GO TO 440
C  TEST WHETHER THE MAXIMAL NUMBER OF ITERATIONS IS REACHED.
        IF(ITER.EQ.MAXIT) GO TO 400
C  CARRY OUT ONE MORE STEP OF THE ITERATION PROCESS.
        P2 = P
```

```
        F2 = FPMS
        IF(ICHECK.NE.0) GO TO 340
        IF((F2-F3).GT.ACC) GO TO 335
C  OUR INITIAL CHOICE OF P IS TOO LARGE.
        P = P*0.1E-02
        P3 = P2
        F3 = F2
        GO TO 350
 335    IF((F1-F2).GT.ACC) GO TO 340
C  OUR INITIAL CHOICE OF P IS TOO SMALL
        TYPE *,"VALUE OF P",P
        P = P*0.1E+04
        P1 = P2
        F1 = F2
        GO TO 350
C  TEST WHETHER THE ITERATION PROCESS PROCEEDS AS THEORETICALLY
C  EXPECTED.
 340    IF(F2.GE.F1 .OR. F2.LE.F3) GO TO 410
        ICHECK = 1
C  FIND THE NEW VALUE FOR P.
        P = RATION(P1,F1,P2,F2,P3,F3)
 350  CONTINUE
C  ERROR CODES AND MESSAGES.
 400  IER = 3
      GO TO 440
 410  IER = 2
      GO TO 440
 420  IER = 1
      GO TO 440
 430  IER = -1
 440  RETURN
      END
      SUBROUTINE BSPLIN(T,N,K,X,L,H)
C  SUBROUTINE BSPLIN EVALUATES THE (K+1) NON-ZERO B-SPLINES OF
C  DEGREE K AT T(L) <= X < T(L+1) USING THE STABLE RECURRENCE
C  RELATION OF DE BOOR AND COX.
C  THE DIMENSION SPECIFICATIONS OF THE FOLLOWING ARRAYS MUST BE
C  AT LEAST H(K+1),HH(K).
      DIMENSION T(N),H(6),HH(5)
      H(1) = 1.
      DO 20 J=1,K
        DO 10 I=1,J
          HH(I) = H(I)
  10    CONTINUE
        H(1) = 0.
        DO 20 I=1,J
          LI = L+I
          LJ = LI-J
          F = HH(I)/(T(LI)-T(LJ))
          H(I) = H(I)+F*(T(LI)-X)
          H(I+1) = F*(X-T(LJ))
  20  CONTINUE
      RETURN
      END
      SUBROUTINE COSSIN(PIV,WI,WW,COS,SIN)
C  SUBROUTINE COSSIN CALCULATES THE PARAMETERS OF A GIVENS
C  TRANSFORMATION WITHOUT SQUARE ROOTS.
      STORE = PIV*WI
      DD = WW+STORE*PIV
      IF(ABS(DD).LT.1.E-36) DD=1.E-36
      COS = WW/DD
```

```
      SIN = STORE/DD
      WW = DD
      WI = COS*WI
      RETURN
      END
      SUBROUTINE ROTATE(PIV,COS,SIN,A,B)
C  SUBROUTINE ROTATE APPLIES A GIVENS ROTATION TO A AND B.
      STORE = B
      B = COS*STORE+SIN*A
      A = A-PIV*STORE
      RETURN
      END
      SUBROUTINE BACK(A,Z,N,K,C)
C  SUBROUTINE BACK CALCULATES THE SOLUTION OF THE SYSTEM OF
C  EQUATIONS A*C = Z WITH A A N X N UNIT UPPER TRIANGULAR MATRIX
C  OF BANDWIDTH K+1.
C  ATTENTION: THE FIRST DIMENSION SPECIFICATION OF MATRIX A MUST
C  BE THE SAME AS IN THE CALLING PROGRAM.
      DIMENSION A(200,K),Z(N),C(N)
      C(N) = Z(N)
      I = N-1
      IF(I.EQ.0) GO TO 30
      DO 20 J=2,N
        STORE = Z(I)
        I1 = K
        IF(J.LE.K) I1 = J-1
        M = I
        DO 10 L=1,I1
          M = M+1
          STORE = STORE-C(M)*A(I,L)
 10     CONTINUE
        C(I) = STORE
        I = I-1
 20   CONTINUE
 30   RETURN
      END
      SUBROUTINE NKNOT(X,M,T,N,FPINT,NRDATA,NRINT)
C  SUBROUTINE NKNOT LOCATES AN ADDITIONAL KNOT FOR A SPLINE OF DEGREE
C  K AND ADJUSTS THE CORRESPONDING PARAMETERS,I.E.
C    T      : THE POSITION OF THE KNOTS.
C    N      : THE NUMBER OF KNOTS.
C    NRINT : THE NUMBER OF KNOTINTERVALS.
C    FPINT : THE SUM OF SQUARES OF RESIDUAL RIGHT HAND SIDES
C            FOR EACH KNOT INTERVAL.
C    NRDATA: THE NUMBER OF DATA POINTS INSIDE EACH KNOT INTERVAL.
C  THE ARRAYS T,FPINT AND NRDATA MUST HAVE THE SAME DIMENSION
C  SPECIFICATIONS AS IN THE CALLING PROGRAM.
      DIMENSION X(M),T(200),FPINT(200),NRDATA(200)
      K = (N-NRINT-1)/2
C  SEARCH FOR KNOT INTERVAL T(NUMBER+K) <= X <= T(NUMBER+K+1) WHERE
C  FPINT(NUMBER) IS MAXIMAL ON THE CONDITION THAT NRDATA(NUMBER)
C  NOT EQUALS ZERO.
      FPMAX = 0.
      JBEGIN = 1
      DO 20 J=1,NRINT
        JPOINT = NRDATA(J)
        IF(FPMAX.GE.FPINT(J) .OR. JPOINT.EQ.0) GO TO 10
        FPMAX = FPINT(J)
        NUMBER = J
        MAXPT = JPOINT
        MAXBEG = JBEGIN
```

```
   10     JBEGIN = JBEGIN+JPOINT+1
   20  CONTINUE
C  LET COINCIDE THE NEW KNOT T(NUMBER+K+1) WITH A DATA POINT X(NRX)
C  INSIDE THE OLD KNOT INTERVAL T(NUMBER+K) <= X <= T(NUMBER+K+1).
       IHALF = MAXPT/2+1
       NRX = MAXBEG+IHALF
       NEXT = NUMBER+1
       IF(NEXT.GT.NRINT) GO TO 40
C  ADJUSTS THE DIFFERENT PARAMETERS.
       DO 30 J=NEXT,NRINT
         JJ = NEXT+NRINT-J
         FPINT(JJ+1) = FPINT(JJ)
         NRDATA(JJ+1) = NRDATA(JJ)
         JK = JJ+K
         T(JK+1) = T(JK)
   30  CONTINUE
   40  NRDATA(NUMBER) = IHALF-1
       NRDATA(NEXT) = MAXPT-IHALF
       FPINT(NUMBER) = FPMAX*FLOAT(NRDATA(NUMBER))/FLOAT(MAXPT)
       FPINT(NEXT) = FPMAX*FLOAT(NRDATA(NEXT))/FLOAT(MAXPT)
       JK = NEXT+K
       T(JK) = X(NRX)
       N = N+1
       NRINT = NRINT+1
       RETURN
       END
       SUBROUTINE DISCO(T,N,K2,B)
C  SUBROUTINE DISCO CALCULATES THE DISCONTINUITY JUMPS OF THE KTH
C  DERIVATIVE OF THE B-SPLINES OF DEGREE K AT THE KNOTS T(K+2)..T(N-K-1)
C  THE FIRST DIMENSION SPECIFICATION OF THE MATRIX B MUST BE THE SAME AS
C  IN THE CALLING PROGRAM; H MUST HAVE A DIMENSION SPECIFICATION AT
C  LEAST 2*K+2.
       DIMENSION T(N),B(200,K2),H(12)
       K1 = K2-1
       K = K1-1
       NK1 = N-K1
       DO 40 L=K2,NK1
         LMK = L-K1
         DO 10 J=1,K1
           IK = J+K1
           LJ = L+J
           LK = LJ-K2
           H(J) = T(L)-T(LK)
           H(IK) = T(L)-T(LJ)
   10    CONTINUE
         LP = LMK
         DO 30 J=1,K2
           JK = J+K
           PROD = 1.
           DO 20 I=J,JK
             PROD = PROD*H(I)
   20      CONTINUE
           LK = LP+K1
           B(LMK,J) = (T(LK)-T(LP))/PROD
           LP = LP+1
   30    CONTINUE
   40  CONTINUE
       RETURN
       END
       FUNCTION RATION(P1,F1,P2,F2,P3,F3)
C  GIVEN THREE POINTS (P1,F1),(P2,F2) AND (P3,F3), FUNCTION RATION
```

```
C  GIVES THE VALUE OF P SUCH THAT THE RATIONAL INTERPOLATING FUNCTION
C  OF THE FORM R(P) = (U*P+V)/(P+W) EQUALS ZERO AT P.
      IF(P3.GT.0.) GO TO 10
C  VALUE OF P IN CASE P3 = INFINITY.
      P = (P1*(F1-F3)*F2-P2*(F2-F3)*F1)/((F1-F2)*F3)
      GO TO 20
C  VALUE OF P IN CASE P3 ~= INFINITY.
  10  H1 = F1*(F2-F3)
      H2 = F2*(F3-F1)
      H3 = F3*(F1-F2)
      P = -(P1*P2*H3+P2*P3*H1+P3*P1*H2)/(P1*H1+P2*H2+P3*H3)
C  ADJUST THE VALUE OF P1,F1,P3 AND F3 SUCH THAT F1 > 0 AND F3 < 0.
  20  IF(F2.LT.0.) GO TO 30
      P1 = P2
      F1 = F2
      GO TO 40
  30  P3 = P2
      F3 = F2
  40  RATION = P
      RETURN
      END
      FUNCTION DERIV(T,N,C,NK1,NU,ARG,L)
C  FUNCTION DERIV EVALUATES A SPLINE S(X) OF DEGREE K WHICH IS
C  GIVEN IN ITS NORMALIZED B-SPLINE REPRESENTATION OR CALCULATES
C  DERIVATIVES OF ANY SPECIFIED ORDER NU.
C
C  CALLING SEQUENCE
C      VALUE = DERIV(T,N,C,NK1,NU,ARG,L)
C
C  INPUT PARAMETERS:
C    T     : ARRAY,MINIMUM LENGTH N, WHICH CONTAINS THE POSITION
C            OF THE KNOTS OF S(X),I.E. THE POSITION OF THE INTERIOR
C            KNOTS T(K+2),...T(N-K-1) AS WELL AS THE POSITION OF THE
C            KNOTS T(1),...T(K+1) AND T(N-K),...T(N) WHICH ARE NEEDED
C            FOR THE B-SPLINE REPRESENTATION.
C    N     : INTEGER VALUE GIVING THE TOTAL NUMBER OF KNOTS OF S(X).
C    C     : ARRAY,LENGTH NK1, CONTAINING THE B-SPLINE COEFFICIENTS.
C    NK1   : INTEGER VALUE,GIVING THE DIMENSION OF S(X),I.E.NK1 = N-K-1.
C    NU    : INTEGER VALUE WHICH GIVES THE ORDER OF THE DERIVATIVE.
C    ARG   : REAL VALUE,GIVING THE VALUE OF THE ARGUMENT.
C    L     : INTEGER VALUE WHICH SPECIFIES THE POSITION OF THE ARGUMENT
C            I.E.    T(L) <= ARG <  T(L+1)    OR
C                    L = NK1 IF ARG = T(NK1+1).
C
C  OUTPUT PARAMETER:
C    VALUE: REAL VALUE,GIVING THE VALUE OF THE NUTH DERIVATIVE OF
C            S(X) AT X = ARG.
C
C  OTHER SUBROUTINES REQUIRED: NONE.
C
C  RESTRICTIONS:
C    1) NU >= 0
C    2) T(K+1) <= ARG <= T(NK1+1)
C  THE DIMENSION SPECIFICATION OF THE ARRAY H MUST BE AT LEAST K+1.
      DIMENSION T(N),C(NK1),H(6)
      DERIV = 0.
      K1 = N-NK1
      IF(NU.LT.0 .OR. NU.GE.K1) RETURN
      DO 100 I=1,K1
        IK = L+I-K1
        H(I) = C(IK)
```

```
100   CONTINUE
      IF(NU.EQ.0) GO TO 300
      NU1 = NU+1
      DO 200 J=2,NU1
        DO 200 JJ=J,K1
          I = J+K1-JJ
          LI = L+I-K1
          LJ = L+I-J+1
          H(I) = (H(I)-H(I-1))/(T(LJ)-T(LI))
200   CONTINUE
      IF(NU.EQ.K1-1) GO TO 500
300   NU2 = NU+2
      DO 400 J=NU2,K1
        DO 400 JJ=J,K1
          I = J+K1-JJ
          LI = L+I-K1
          LJ = L+I-J+1
          H(I) = ((ARG-T(LI))*H(I)+(T(LJ)-ARG)*H(I-1))/(T(LJ)-T(LI))
400   CONTINUE
500   DERIV = H(K1)
      IF(NU.EQ.0) RETURN
      DO 600 I=1,NU
        DERIV = DERIV*FLOAT(K1-I)
600   CONTINUE
      RETURN
      END
```

APPENDIX D

THE PROGRAM TO FIND B-SPLINE COEFFICIENT

```
PROGRAM BSPLINE(INPUT,OUTPUT,INFILE,OUTFILE);

  TYPE
    BYTE = 0..255;
    IMAGEROW1 = PACKED ARRAY [0..255] OF BYTE;
    ROW1 = PACKED ARRAY [0..257] OF BYTE;
    SHIP = PACKED ARRAY [0..63,0..255] OF BYTE;
    DA1 = PACKED ARRAY [0..512] OF REAL;
    DA2 = PACKED ARRAY [1..300] OF REAL;
  VAR
    R,C,F0,F1,F2,F3,F4,F5,F6,F7,F8  :  BYTE;
    F : ARRAY [0..65] OF ROW1;
    A,IMAGE : SHIP;
    INFILE : FILE OF IMAGEROW1;
    SPX,SPY,SPX1,SPY1 : PACKED ARRAY [0..512] OF REAL;
    R1,C1,R2,C2,COIR,I,J,J1,COUNT,NEG :INTEGER;
    TEMPX,TEMPY,RA : REAL;
    M,IOPT,K,IPAR,N,IER,ANS,NU,ANS1,ANS2,ANS3 : INTEGER;
    NK1,NEND,L,MET :INTEGER;
    W,Z :PACKED ARRAY [0..512] OF REAL;
    S,ZB,ZE,FP : REAL;
    ARG,THETA,TOLE : REAL;
    FLAG_BEGIN,FLAG_END,AK,LUMP : INTEGER;
    BEG,EN : PACKED ARRAY [1..5] OF REAL;
    BEGN,ENN : PACKED ARRAY [1..5] OF INTEGER;
    T,CX,CY :PACKED ARRAY [1..300] OF REAL;
    COL,ROW :PACKED ARRAY [0..512] OF REAL;
    X,Y :PACKED ARRAY [0..512] OF REAL;
    DCY,DCX,ARE,CY1,DMAX,DMIN : REAL;
    CYMIN,CYMAX,MAXCY : PACKED ARRAY [1..5] OF REAL;
    AREA : PACKED ARRAY [1..5] OF REAL;
    OUTFILE : FILE OF IMAGEROW1;
    NAME : PACKED ARRAY [1..20] OF CHAR;
    PEAK,STAR,TER :PACKED ARRAY [1..5] OF REAL;

 (* FILTER THE POINTS *)

    PROCEDURE STORE;
    BEGIN
      TEMPX:=C-1; TEMPY:=64-(R-1);
      IF I>0 THEN BEGIN
        FOR J:=0 TO M DO BEGIN
          IF(COL[J]=TEMPX) AND (ROW[J]=TEMPY)
          THEN I:=J;
        END;
        COL[I] :=TEMPX;
        ROW[I] :=TEMPY;
        M:=I;
        I:=I+1;
      END
      ELSE BEGIN
        COL[I] :=TEMPX;
        ROW[I] :=TEMPY;
        M:=0;
        I:=1;
      END;
    END;
```

```
PROCEDURE CMOVE;
BEGIN
  F0 :=F[R-1,C]; F1:=F[R-1,C+1]; F2:=F[R,C+1];
  F3:=F[R+1,C+1]; F4:=F[R+1,C]; F5:=F[R+1,C-1];
  F6:=F[R,C-1]; F7:=F[R-1,C-1];
  CASE COIR OF
    0: BEGIN
      IF (F3=0) AND (F2=255) THEN BEGIN C:=C+1;
       COIR:=1;
      END
      ELSE IF (F2=0) AND (F1=255) THEN BEGIN R:=R-1;
        STORE; C:=C+1; COIR:=0; END
      ELSE IF (F1=0) AND (F0=255) THEN BEGIN R:=R-1;
        COIR:=0; END
      ELSE IF (F0=0) AND (F7=255) THEN BEGIN C:=C-1;
        STORE; R:=R-1; COIR:=3; END
      ELSE IF (F7=0) AND (F6=255) THEN BEGIN C:=C-1;
        COIR:=3; END
      ELSE IF (F6=0) AND (F5=255) THEN BEGIN R:=R+1;
        STORE; C:=C-1; COIR:=3; END
      ELSE BEGIN R:=R+1; COIR:=2; END;
    END;
    1: BEGIN
      IF (F5=0) AND (F4=255) THEN BEGIN R:=R+1;
        COIR:=2; END
      ELSE IF (F4=0) AND (F3=255) THEN BEGIN C:=C+1;
        STORE; R:=R+1; COIR:=1; END
      ELSE IF (F3=0) AND (F2=255) THEN BEGIN C:=C+1;
        COIR:=1; END
      ELSE IF (F2=0) AND (F1=255) THEN BEGIN R:=R-1;
        STORE; C:=C+1; COIR:=0; END
      ELSE IF (F1=0) AND (F0=255) THEN BEGIN R:=R-1;
        COIR:=0; END
      ELSE IF (F0=0) AND (F7=255) THEN BEGIN C:=C-1;
        STORE; R:=R-1; COIR:=0; END
      ELSE BEGIN C:=C-1; COIR:=3; END;
    END;
    2: BEGIN
      IF (F7=0) AND (F6=255) THEN BEGIN C:=C-1;
        COIR:=3; END
      ELSE IF (F6=0) AND (F5=255) THEN BEGIN R:=R+1;
        STORE; C:=C-1; COIR:=3; END
      ELSE IF (F5=0) AND (F4=255) THEN BEGIN R:=R+1;
        COIR:=2; END
      ELSE IF (F4=0) AND (F3=255) THEN BEGIN C:=C+1;
        STORE; R:=R+1; COIR:=1; END
      ELSE IF (F3=0) AND (F2=255) THEN BEGIN C:=C+1;
        COIR:=1; END
      ELSE IF (F2=0) AND (F1=255) THEN BEGIN R:=R-1;
        STORE; C:=C+1; COIR:=1; END
      ELSE BEGIN R:=R-1; COIR:=0; END;
    END;
    3: BEGIN
      IF (F1=0) AND (F0=255) THEN BEGIN R:=R-1;
        COIR:=0; END
      ELSE IF (F0=0) AND (F7=255) THEN BEGIN C:=C-1;
```

```
      STORE; R:=R-1; CDIR:=0; END
    ELSE IF (F7=0) AND (F6=255) THEN BEGIN C:=C-1;
      CDIR:=3; END
    ELSE IF (F6=0) AND (F5=255) THEN BEGIN R:=R+1;
      STORE; C:=C-1; CDIR:=2; END
    ELSE IF (F5=0) AND (F4=255) THEN BEGIN R:=R+1;
      CDIR:=2; END
    ELSE IF (F4=0) AND (F3=255) THEN BEGIN C:=C+1;
      STORE; R:=R+1; CDIR:=2; END
    ELSE BEGIN C:=C+1; CDIR:=1; END;
   END;
  END;
END;


PROCEDURE INITIAL;
  BEGIN
  FOR I :=0 TO 255 DO BEGIN
    X[I] :=0;
    Y[I] :=0;
  END;
  I:=0;
END;


PROCEDURE FIRST;
  VAR
    N,M :INTEGER;
BEGIN
  N := 0; M:=0;
  FOR C :=0 TO 200 DO BEGIN
    FOR R :=0 TO 63 DO BEGIN
      IF (F[R,C]=255) AND (N=0) THEN BEGIN
        C1 := C; R1 :=R; N := N+1;
      END;
      IF (F[R,255-C]=255) AND (M=0) THEN BEGIN
        C2 := 255-C; R2 := R; M :=M+1;
      END;
    END;
  END;
END;

PROCEDURE ROTATE;
BEGIN
  NEG:=0;
  IF R1=R2 THEN  BEGIN THETA:=0.0;
    FOR I:=0 TO M DO BEGIN
      X[I]:=COL[I]; Y[I]:=ROW[I];
    END;
  END
  ELSE THETA:=ABS(ARCTAN((R2-R1)/(C2-C1)));
  IF (THETA<>0) AND (R2>R1) THEN BEGIN
    FOR I:=0 TO M DO BEGIN
      X[I]:=COL[I]*COS(THETA)-ROW[I]*SIN(THETA);
      Y[I]:=COL[I]*SIN(THETA)+ROW[I]*COS(THETA);
      IF Y[I]>=63.0 THEN NEG:=NEG+1;
    END;
```

```
      END
      ELSE IF (THETA<>0) AND (R2<R1) THEN BEGIN
        FOR I:=0 TO M DO BEGIN
          X[I]:=COL[I]*COS(THETA)+ROW[I]*SIN(THETA);
          Y[I]:=-COL[I]*SIN(THETA)+ROW[I]*COS(THETA);
          IF Y[I]>=63.0 THEN NEG:=NEG+1;
        END;
      END;
(* FILTER *)
      I:=0;
      FOR K:=0 TO M DO BEGIN
        TEMPX:=X[K]; TEMPY:=Y[K];
        IF I>0 THEN BEGIN
          FOR J:=0 TO M DO BEGIN
            IF (X[J]=TEMPX) AND (Y[J]=TEMPY) THEN I:=J;
          END;
          X[I]:=TEMPX; Y[I]:=TEMPY; M:=I; I:=I+1;
        END
        ELSE BEGIN X[I]:=TEMPX; Y[I]:=TEMPY; M:=0; I:=1;
        END;
      END;
    END;

    PROCEDURE PARAM( X:DA1; Y:DA1; VAR Z:DA1; W:DA1;
      M:INTEGER; VAR ZB:REAL; VAR ZE:REAL; K:INTEGER;
      S:REAL; VAR N:INTEGER; VAR T:DA2; VAR CX:DA2;
      VAR CY:DA2; VAR FP:REAL; IOPT:INTEGER;
      IPAR:INTEGER; VAR IER:INTEGER); FORTRAN;
    PROCEDURE INITT(SPEED:INTEGER); FORTRAN;
    PROCEDURE VWINDO(XMIN:REAL; XRANGE:REAL; YMIN:REAL;
      YRANGE:REAL); FORTRAN;
    PROCEDURE MOVEA(X:REAL; Y:REAL); FORTRAN;
    PROCEDURE DRAWA(X:REAL; Y:REAL); FORTRAN;
    PROCEDURE ANCHO(ICHAR:INTEGER); FORTRAN;
    PROCEDURE FINITT(I1:INTEGER; I2:INTEGER); FORTRAN;
    PROCEDURE DASHA(X:REAL; Y:REAL; L:INTEGER); FORTRAN;

    FUNCTION DERIV(%REF T:DA2; N:INTEGER; CX:DA2;
      NK1:INTEGER; NU:INTEGER; ARG:REAL;
      L:INTEGER) :REAL; FORTRAN;
    PROCEDURE SPLCOEF;
    BEGIN
      I:=5; LUMP:=0;
      WHILE (I<=N-5) AND (LUMP=0) DO BEGIN
        IF(CY[I-1]>CY[I]) AND (CY[I+1]>CY[I]) AND
          (CY[I+2]>CY[I+1]) THEN
          LUMP:=1;
        I:=I+1;
      END;
      IF LUMP=1 THEN BEGIN
        FLAG_BEGIN:=0; AK:=1; FLAG_END:=0; I:=5;
        WHILE I<=N-5 DO BEGIN
          IF FLAG_BEGIN=0 THEN BEGIN
            IF (CY[I-1]>CY[I]) AND (CY[I+1]>CY[I]) AND
              (CY[I+2]>CY[I+1]) THEN BEGIN
              BEG[AK]:=T[I]; FLAG_BEGIN:=1; BEGN[AK]:=I;
            END;
```

```
      END
      ELSE IF FLAG_BEGIN=1 THEN BEGIN
        IF (CY[I-1]>=CY[I]) AND (CY[I]>=CY[I+1]) AND
          (CY[I+2]>=CY[I+1]) THEN BEGIN
          EN[AK]:=T[I+1]; FLAG_END:=1; ENN[AK]:=I+1;
        END;
      END;
      IF (FLAG_BEGIN=1) AND (FLAG_END=1) THEN BEGIN
        FLAG_BEGIN:=0; FLAG_END:=0; AK:=AK+1; I:=I-1;
      END;
      I:=I+1;
    END;
  END
  ELSE IF LUMP=0 THEN BEGIN
    FLAG_BEGIN:=0; FLAG_END:=0; I:=5; AK:=1;
    WHILE I<=N-5 DO BEGIN
      IF FLAG_BEGIN=0 THEN BEGIN
        IF(CY[I-1]>=CY[I]) AND (CY[I+1]>CY[I]) AND
          (CY[I+2]>CY[I]) THEN BEGIN
          BEG[AK]:=T[I]; FLAG_BEGIN:=1; BEGN[AK]:=I;
        END;
      END
      ELSE IF FLAG_BEGIN=1 THEN BEGIN
        IF(CY[I-1]>=CY[I]) AND ((CY[I]<=CY[I+1]+TOLE)
          AND (CY[I]>=CY[I+1]-TOLE)) AND
          ((CY[I]<=CY[I+2]+TOLE) AND
          (CY[I]>=CY[I+2]+TOLE)) THEN BEGIN
          EN[AK]:=T[I]; FLAG_END:=1; ENN[AK]:=I;
        END;
      END;
      IF (FLAG_BEGIN=1) AND (FLAG_END=1) THEN BEGIN
        FLAG_BEGIN:=0; FLAG_END:=0; AK:=AK+1; I:=I-1;
      END;
      I:=I+1;
    END;
    IF FLAG_BEGIN=1 THEN BEGIN
      I:=BEGN[AK];
      WHILE (I<=N-5) AND (FLAG_END=0) DO BEGIN
        IF (CY[I-1]>=CY[I]) AND
          ((CY[I]<=CY[I+1]+TOLE)
          AND (CY[I]>=CY[I+1]-TOLE)) THEN BEGIN
          EN[AK]:=T[I]; ENN[AK]:=I; FLAG_END:=1;
        END;
        I:=I+1;
      END;
    END;
  END;
END;

PROCEDURE ALUMP;
BEGIN
  AK:=1;
  WHILE (ENN[AK])>0 DO BEGIN
    AREA[AK]:=0; CYMAX[AK]:=0.0; CYMIN[AK]:=1000.0;
    FOR I:=(BEGN[AK]) TO (ENN[AK]) DO BEGIN
      IF CYMAX[AK] <= CY[I] THEN BEGIN
        CYMAX[AK]:=CY[I]; MAXCY[AK]:=T[I]
```

```
          END;
          IF CYMIN[AK] >= CY[I] THEN
            CYMIN[AK]:=CY[I];
        END;
        CY1:=CY[BEGN[AK]]-CYMIN[AK];
        FOR I:=(BEGN[AK]) TO (ENN[AK]-1) DO BEGIN
          DCY:=CY[I+1]-CY[I];
          DCX:=CX[I+1]-CX[I];
          AREA[AK]:=AREA[AK]+CY1*DCX+DCY*DCX/2.0;
          WRITELN('AREA=',AREA[AK],' I=',I,' AK=',AK);
          CY1:=CY1+DCY;
        END;
        AK:=AK+1;
      END;
    END;
    PROCEDURE POSINX;
    BEGIN
      I:=0; MET:=0;
      WHILE (I<=M-1) AND (MET=0) DO BEGIN
        IF TEMPX=Z[I] THEN BEGIN
          TEMPY:=X[I]; MET:=1;
        END;
        I:=I+1;
      END;
    END;

(*********************************)
(* MAIN PROGRAM *)
    BEGIN
      WRITELN('INPUT CUT OR CONLINE FILENAME');
      READLN(NAME);
      OPEN (INFILE,NAME,HISTORY := OLD,
        ACCESS_METHOD :=SEQUENTIAL,
        RECORD_LENGTH :=256,RECORD_TYPE :=FIXED);
      RESET(INFILE);
      R :=0;
      WHILE NOT EOF (INFILE) DO
      BEGIN
        READ (INFILE,IMAGE[R]);
        FOR C := 0 TO 255 DO
          F[R+1,C+1] := IMAGE[R,C];
        R := R+1;
      END;
      CLOSE(INFILE);
      FIRST;
      R :=R1; C :=C1; CDIR :=1;
      INITIAL;
      WHILE (C<>C2) DO BEGIN
        STORE;
        CMOVE;
      END;
      ROTATE;
      FOR I:=0 TO M DO BEGIN
        IF NEG<>0 THEN  Y[I]:=Y[I]-20.0;
      END;
      M:=M+1; K:=2; S:=0.1; IOPT:=0; IPAR:=0;
      FOR I:=0 TO M DO
```

```
  W[I]:=1.0;
ANS:=1;
WHILE ANS=1 DO BEGIN
  WRITELN('OLD S=',S,'NEW S=');
  READ(S);
  WRITELN('OLD K=',K,'K=');
  READ(K);
  PARAM(X,Y,Z,W,M,ZB,ZE,K,S,N,T,CX,CY,FP,IOPT,
    IPAR,IER);
  WRITELN(' S=',S,' IER=',IER,' M=',M,' N=',N,
    ' CX[N]=',CX[N], ' CY[N-4]=',CY[N-4]);
  WRITELN(' PRINTING  YES=1 PLOT X-Y=2,CX,CY=3 ',
    'X-Y-CX-CY=4',' NO=5');
  READ(ANS1);
  IF ANS1=1 THEN BEGIN
    FOR I:=1 TO N DO
      WRITELN('CX=',CX[I],'    CY=',CY[I],
        '    T=',T[I]);
    END
  ELSE IF ANS1=2 THEN BEGIN
    INITT(960);
    VWINDO(0.0,Z[M-1],0.0,256.0);
    MOVEA(Z[0],X[0]);
    FOR I:=0 TO M-1 DO
      DRAWA(Z[I],X[I]);
    MOVEA(Z[0],Y[0]);
    FOR I:=0 TO M-1 DO
      DASHA(Z[I],Y[I],2);
    FINITT(0,0);
  END
  ELSE IF ANS1=4 THEN BEGIN
    INITT(960);
    VWINDO(0.0,Z[M-1],0.0,256.0);
    MOVEA(Z[0],X[0]);
    FOR I:=0 TO M-1 DO
      DRAWA(Z[I],X[I]);
    MOVEA(Z[0],Y[0]);
    FOR I:=0 TO M-1 DO
      DRAWA(Z[I],Y[I]);
    MOVEA(T[4],CX[4]);
    FOR I:=4 TO N-4 DO
      DASHA(T[I],CX[I],2);
    FOR I:=4 TO N-4 DO BEGIN
      MOVEA(T[I]-1.5,CX[I]-1.5);
      ANCHO(111);
    END;
    MOVEA(T[4],CY[4]);
    FOR I:=4 TO N-4 DO
      DASHA(T[I],CY[I],2);
    FOR I:=4 TO N-4 DO BEGIN
      MOVEA(T[I]-1.5,CY[I]-1.5);
      ANCHO(111);
    END;
    DMIN:=0.0; DMAX:=256.0; TEMPX:=DMIN;
    WHILE TEMPX<=Z[M-1] DO BEGIN
      MOVEA(TEMPX,DMIN);
      DRAWA(TEMPX,DMIN);
```

```
      DRAWA(TEMPX,DMAX);
      TEMPX:=TEMPX+10.0;
    END;
    DMIN:=0.0; DMAX:=Z[M-1]; TEMPX:=DMIN;
    WHILE TEMPX<=256.0 DO BEGIN
      MOVEA(DMIN,TEMPX);
      DRAWA(DMIN,TEMPX);
      DRAWA(DMAX,TEMPX);
      TEMPX:=TEMPX+10.0;
    END;
    FINITT(0,0);
  END
  ELSE IF ANS1=3 THEN BEGIN
    INITT(960);
    VWINDO(0.0,T[N],0.0,256.0);
    MOVEA(T[4],CX[4]);
    FOR I:=4 TO N-4 DO
      DRAWA(T[I],CX[I]);
    FOR I:=4 TO N-4 DO BEGIN
      MOVEA(T[I]-1.5,CX[I]-1.5);
      ANCHO(111);
    END;
    MOVEA(T[4],CY[4]);
    FOR I:=4 TO N-4 DO
      DASHA(T[I],CY[I],2);
      FOR I:=4 TO N-4 DO BEGIN
        MOVEA(T[I]-1.5,CY[I]-1.5);
        ANCHO(111);
      END;
    FINITT(0,0);
  END;
(* IMPORTANT FORTRAN DECLEAR FROM 1 *)
  NK1:=N-K-1; L:=K+1; NEND:=N-K; J1:=0;
  FOR I:=L TO NEND DO BEGIN
    ARG:=T[I];
    WHILE(ARG>=T[L+1])  AND (L<NK1) DO
      L:=L+1;
    SPX1[J1]:= DERIV(T,N,CX,NK1,0,ARG,L);
    SPY1[J1]:= DERIV(T,N,CY,NK1,0,ARG,L);
    J1:=J1+1;
  END;
  J:=0; L:=K+1;
  FOR I:=L TO NEND DO BEGIN
    ARG:=T[I];
  WHILE (ARG>=T[L+1]) AND (L<NK1) DO
    L:=L+1;
  TEMPX:=T[I+1]-ARG;
    IF TEMPX>=7.0 THEN BEGIN
      WHILE ARG<T[I+1] DO BEGIN
        SPX[J] := DERIV(T,N,CX,NK1,0,ARG,L);
        SPY[J] := DERIV(T,N,CY,NK1,0,ARG,L);
        ARG:=ARG+2.0; J:=J+1;
      END;
    END
    ELSE BEGIN
      SPX[J] := DERIV(T,N,CX,NK1,0,ARG,L);
      SPY[J] := DERIV(T,N,CY,NK1,0,ARG,L);
```

```
      J:=J+1;
    END;
  END;
  WRITELN('DO YOU WANT PLOTING YES=1 ',
    'PRINT=2 COMPARE=3 NO=4');
  READ(ANS2);
  IF ANS2=1 THEN BEGIN
    INITT(960);
    VWINDO(0.0,256.0,0.0,256.0);
    MOVEA(X[0],Y[0]);
    FOR I:=0 TO M-1 DO
      DRAWA(X[I],Y[I]);
    FOR I:=0 TO J1-1 DO BEGIN
      MOVEA(SPX1[I]-1.5,SPY1[I]-1.5);
      ANCHO(111);
    END;
    MOVEA(SPX[0],SPY[0]);
    FOR I:=1 TO J-1 DO
      DASHA(SPX[I],SPY[I],2);
    FINITT(0,0);
  END
  ELSE IF ANS2=2 THEN BEGIN
    FOR I:=0 TO J1-1 DO
      WRITELN('SPX1=',SPX1[I],
        '    SPY1=',SPY1[I]);
  END
  ELSE IF ANS2=3 THEN BEGIN
    WRITELN('TOL=');
    READ(TOLE);
    SPLCOEF;
    ALUMP;
    AK:=1; RA:=X[M-1]-X[0];
    WHILE EN[AK]>0 DO BEGIN
      TEMPX:=BEG[AK]; POSINX;
      STAR[AK]:=((TEMPY-X[0])-RA/2)/RA;
      TEMPX:=EN[AK]; POSINX;
      TER[AK]:=((TEMPY-X[0])-RA/2)/RA;
      TEMPX:=MAXCY[AK]; POSINX;
      PEAK[AK]:=((TEMPY-X[0])-RA/2)/RA;
      AREA[AK] :=(AREA[AK])/(RA**2);
      WRITELN('BEGIN=',STAR[AK],' END=',
        TER[AK],' TOTAL=',RA,
        ' AREA=',AREA[AK],
        ' PEAK=',PEAK[AK]);
      AK:=AK+1;
    END;
  END;
  WRITELN('DO YOU WANT RUN AGAIN YES=1',
    ' NO=2');
  READ(ANS);
  IF ANS=1 THEN BEGIN
    WRITELN('IOPT=');
    READ(IOPT);
  END;
END; (* WHILE *)
END.
```

LIST OF REFERENCES

1. Hall, Ernest L., Computer Image Processing and Recognition, pp. 403-409, Academic Press, 1979.

2. Ballard, H. Dana and Brown, Christopher M., Computer Vision, Prentice-Hall, INC., Englewood Cliffs, New Jersey, 1982.

3. Computer Graphics and Image Processing, Algorithm For Smoothing Data With Periodic And Parametric Spline, By P. Dierckx, October 16, 1981.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Superintendent ATTN: Prof. Harriett Rigas, Code 62Rr Naval Postgraduate School Monterey, California 93943	1
4.	Superintendent ATTN: Prof. Mitchell L. Cotton, Code 62Cc Naval Postgraduate School Monterey, California 93943	1
5.	Superintendent ATTN: Prof. Chin-Hwa Lee, Code 62Le Naval Postgraduate School Monterey, California 93943	6
6.	Lt. Werawong Thavamongkon R.T.N. 78/1 Mu. 7 Soi. Bearing Prakanong Bangna Bangkok (10260), Thailand	3

213147

Thesis
T362
c.1

Thavamongkon
Ship outline feature
selection using B-
spline function.